National Center for Sustainable Transportation
A USDOT University Transportation Center

Research Report

April 2025

Real-Time Large-Scale Ridesharing with Flexible Meeting Points

Maged Dessouky, University of Southern California

Zuhayer Mahtab, University of Southern California

METRANS Transportation Consortium
USC | CSULB

TECHNICAL REPORT DOCUMENTATION PAGE

1. Report No. NCST-USC-RR-25-08	**2. Government Accession No.** N/A	**3. Recipient's Catalog No.** N/A
4. Title and Subtitle Real-Time Large-Scale Ridesharing with Flexible Meeting Points		**5. Report Date** April 2025 **6. Performing Organization Code** N/A
7. Author(s) Maged Dessouky, Ph.D., https://orcid.org/0000-0002-9630-6201 Zuhayer Mahtab, https://orcid.org/0000-0003-4793-2705		**8. Performing Organization Report No.** N/A
9. Performing Organization Name and Address University of Southern California METRANS Transportation Consortium University Park Campus, VKC 367 MC:0626 Los Angeles, California 90089-0626		**10. Work Unit No.** N/A **11. Contract or Grant No.** USDOT Grant 69A3552348319 and 69A3552344814
12. Sponsoring Agency Name and Address U.S. Department of Transportation Office of the Assistant Secretary for Research and Technology 1200 New Jersey Avenue, SE, Washington, DC 20590		**13. Type of Report and Period Covered** Final Research Report (January 2024-December 2024) **14. Sponsoring Agency Code** USDOT OST-R
15. Supplementary Notes DOI: https://doi.org/10.7922/G23B5XHG Dataset DOI: https://dx.doi.org/10.6084/m9.figshare.28119947		
16. Abstract In this report, the authors propose an online and large-scale rideshare system that can dynamically match passenger requests with drivers and provide efficient routes to the drivers. The authors developed a greedy insertion-based routing procedure to route thousands of requests in an hour. They incorporated flexible meeting point selection into the framework, which can reduce travel distances for both drivers and passengers. The authors implemented an online incentive and cost-sharing system that can incentivize drivers and passengers for their ride time limit violations and share the cost of a rideshare trip among the passengers fairly. The authors incorporated a request prediction and detour mechanism into the ridesharing framework. To get the most updated travel time and study the effects of ridesharing in a road network, the authors also incorporate a simulation approach into the framework. Numerical experiments performed on the New York Taxicab dataset and a rural dataset based on Kern and Tulare Counties, California, show that the proposed framework is effective, matching thousands of requests per hour. Results also show that ridesharing can cost significantly less compared to ride-hailing services such as Uber or Lyft, and incorporating flexible meeting points can reduce travel distance by 4% on average. Simulation studies show that ridesharing can reduce total vehicle miles traveled by 13% in Manhattan on average. The proposed framework can help transportation officials design real-time and city-scale rideshare systems to alleviate traffic congestion problems in California.		
17. Key Words Ridesharing, Online Systems, Large-Scale Optimization, Simulation		**18. Distribution Statement** No restrictions.

19. Security Classif. (of this report) Unclassified	**20. Security Classif. (of this page)** Unclassified	**21. No. of Pages** 51	**22. Price** N/A

Form DOT F 1700.7 (8-72) Reproduction of completed page authorized

Acknowledgments

This study was funded, partially or entirely, by a grant from the National Center for Sustainable Transportation (NCST), supported by the U.S. Department of Transportation (USDOT) through the University Transportation Centers program. The authors would like to thank the NCST and the USDOT for their support of university-based research in transportation, and especially for the funding provided in support of this project.

Real-Time Large-Scale Ridesharing with Flexible Meeting Points

A National Center for Sustainable Transportation Research Report

Maged Dessouky, Daniel J. Epstein Department of Industrial and Systems Engineering, University of Southern California

Zuhayer Mahtab, Daniel J. Epstein Department of Industrial and Systems Engineering, University of Southern California

Table of Contents

Executive Summary ....
Introduction 1
Literature Review 3
Dynamic Vehicle Routing 3
Dynamic Rideshare Routing Problem 4
Incentive and Shared Cost Mechanisms for Rideshare Systems 5
Future Request Prediction and Simulation Mechanisms 6
Problem Description 8
Proposed Framework 11
Central Coordinator 12
Routing Mechanism 12
Meeting Point Selection Mechanism 16
Incentive and Shared Cost Mechanism 16
Prediction Mechanism 18
Simulation Mechanism 19
Overall Framework 19
Numerical Experiments 21
Description of the Urban Dataset 21
Performance of the Proposed Framework in the NYC Dataset 24
Effect of Flexible Meeting Points 29
Effect of Incentive Budget on the Performance of the Rideshare System 30
Description of the Rural Dataset 32
Performance of the Rideshare System in the Rural Dataset 35
Conclusions 38
References 39
Data Summary 42

List of Tables

Table 1. Parameters and Their Values for the NYC Dataset 24

Table 2. Total Vehicle Distance Traveled, Waiting Time, and Time Loss across All Instances in Solo Driving vs. Ridesharing... 26

Table 3. Passenger Service Rate, Budget Usage Rate, Occupancy Rate, and Trip Distance per Vehicle Across All Instances... 27

Table 4. Average Shared Cost, Incentive Recipient Rate, and Incentive Payment Across All Instances 28

Table 5. Execution Time of Our Proposed Framework Across Different Problem Instances 29

Table 6. Total Travel Distance in Flexible Meeting Points vs. No Meeting Points 30

Table 7. Parameters and Their Values for the Rural Dataset 34

Table 8. Total Vehicle Distance Traveled across All Instances in the Rural Dataset 36

Table 9. Passenger Service Rate, Budget Usage Rate, Average Occupancy Rate, and Average Trip Distance per Rideshare Vehicle Across All Instances in the Rural Dataset 36

Table 10. Average Shared Cost, Incentive Recipient Rate, and the Average Incentive Payment Across All Instances in the Rural Dataset 37

List of Figures

Figure 1. Proposed Framework 11
Figure 2. Algorithm 1: Greedy Insertion 15
Figure 3. Algorithm 2: POCS Mechanism 18
Figure 4. Algorithm 3: Proposed Framework 20
Figure 5. Taxi Zones in Manhattan; source: (New York (N.Y.) Taxi And Limousine Commission, 2019) 23
Figure 6. Traffic Volume Distribution in Manhattan throughout the Day 23
Figure 7. Change in the Percentage of Passengers Served with Change in Budget 31
Figure 8. Change in the Percentage of Budget Used with Change in Budget 32
Figure 9. Map of the Rural Area 33
Figure 10. The Rural Dataset Broken Down into Zones 34

Executive Summary

Many major cities in the United States are plagued by increasing traffic congestion, pollutant emissions, and inaccessibility to transportation. Ridesharing can be a means to solve these problems. This is a system where regular drivers provide rides to other passengers on their way to their destination. Ridesharing can provide flexible and inexpensive rides to passengers and increase transportation accessibility for economically disadvantaged individuals. By efficiently using empty spaces in personal vehicles, rideshare can reduce the need for a personal vehicle, traffic congestion, and pollutant emissions.

For a rideshare system to be effective, passengers need to be matched with the drivers in a way that minimizes driver inconvenience. Therefore, the rideshare system needs to provide the drivers with good quality routes that minimize driver detours, waiting times, and distance traveled. These routes also need to be updated as passengers and drivers enter and exit the rideshare system and as the traffic conditions change. Therefore, the rideshare system needs to react in real time to the changing dynamics of the system. In this report, we propose such a rideshare system. We developed a routing framework based on a greedy insertion heuristic that can efficiently route thousands of requests in an hour with a very limited computation budget.

One way to reduce driver detours is to incorporate flexible meeting points. In this system, a passenger may have to walk to their pickup point from their origin or walk to their destination from the drop-off point. By walking a certain distance, travel distances for both drivers and passengers can be reduced significantly. We incorporated this feature into our rideshare system.

Also, to increase the adoption of the rideshare system, drivers and passengers can be compensated by the platform for their inconveniences. In many cases, providing rides to passengers may result in longer trip times for the drivers and other passengers in the vehicle. This may act as a detrimental factor in the adoption of rideshare. To remedy this and increase the passenger service rate of the rideshare system, we implemented an online incentive and cost-sharing system.

To understand how the rideshare system affects traffic conditions, we incorporated simulation into our rideshare framework. This simulation module provides the rideshare system with the most updated condition of the road network so that the rideshare routes can be robust to changes in traffic conditions. Also, the simulation model facilitates the study of the benefits of introducing a rideshare system into realistic settings.

We conducted numerical experiments on two databases in two different contexts: urban and rural. For the urban area, we chose the New York Taxicab dataset. For the rural area, we constructed a database based on the area surrounding Kern and Tulare Counties, California. Numerical experiments on this dataset showed that our proposed framework could effectively route thousands of rideshare requests in real time. Also, passengers, on average, had to pay only about $1.68 in the urban area and $3.51 in the rural area for a ride. Results also show that incorporating flexible meeting points can reduce the drivers' detours by 4% on average. Results from the simulation study revealed that ridesharing can reduce the total vehicle distance traveled by 13% on average in urban areas and 31% on average in rural areas. Sensitivity analysis of incentive budgets showed that incentivizing rideshare drivers and passengers can be a great way of increasing participation in the rideshare system and can increase passenger service rates significantly.

Introduction

Traffic congestion is an ever-increasing problem in large metropolitan areas in the United States. Commuters lose hours every year stuck in traffic. According to the 2023 Urban Mobility Report, each commuter lost about 54 hours in 2022. In monetary terms, this amounts to $1259 per commuter per year. Nationwide traffic congestion results in 8.5 billion lost hours and 3.3 billion gallons of wasted fuel. Major infrastructure projects have been undertaken to alleviate the problem, such as building freeways or adding more lanes to existing freeways. However, these road expansion projects cannot keep up with the increasing number of personal vehicles. These personal vehicles are also a highly inefficient mode of transportation compared to shared mobility options, as they have a low occupancy rate (Lasley, 2021). Many of the major cities in the U.S. have insufficient public transport options, making transportation inaccessible to economically disadvantaged individuals. Also, in most of the rural areas of the United States, public transportation does not exist.

One way to reduce congestion and pollutant emissions and, at the same time, provide flexible and inexpensive transportation options is ridesharing. This is a system where regular drivers provide commutes to other people on their way to their destination. Since the drivers are regular drivers, their motivation to participate in the system is to reduce travel costs. By sharing the travel costs of the drivers fairly among the passengers, rideshare can provide inexpensive trips to commuters. At the same time, rideshare is more flexible than public transport such as buses or metro. A driver can pick up or drop off a passenger from their location of choice. Since the drivers are regular commuters using their own vehicles, ridesharing does not need a significant investment, as would be true for a mass transit system.

In this study, we make a distinction between ridesharing and ride-hailing. We define ride-hailing as a commercial service where professional drivers pick up and drop off passengers to make a profit. Popular services such as Uber and Lyft are ride-hailing services. Although these services are very popular, they have two disadvantages compared to ridesharing services. First, since the goal of the drivers and the platforms is to make a profit, they will generally charge a passenger more than a ridesharing service. Additionally, these services use surge pricing, where passengers will be charged a lot more in times of high demand. Second, these services add a lot of deadhead miles to a road network, worsening traffic congestion (Z. Li et al., 2021). Many times, these ride-hailing drivers have to drive empty in search of a customer or to go to a customer's origin point. Ridesharing solves these problems. Ridesharing does not add many deadhead miles. If a vehicle is empty of passengers, then the driver can perhaps, with a slight detour, pick up another passenger or continue to their destination. In ridesharing, a passenger has to bear a fraction of the driver's travel cost, making it much cheaper than ride-hailing.

To make ridesharing more appealing to drivers, we need to minimize excessive detours. Therefore, passengers need to be matched with drivers in a way that minimizes inconvenience to drivers. In a practical scenario, a ridesharing system is highly dynamic. Both passengers and drivers can enter or exit the system anytime. Also, traffic conditions in a city are constantly changing. We need to take these changes into account while designing efficient routes for drivers. In this report, we aim to address these issues.

Flexible meeting points can be an effective way to reduce excessive detours. This is a system where the passengers agree to meet the driver at a specified meeting point to be picked up or dropped off. Passengers may have to walk a certain distance to the pickup point or from the drop-off point. Several research papers have been published on this issue, and they show that flexible meeting points can reduce travel times by up to 10% (Fielbaum, 2022; Fielbaum et al., 2021; X. Li et al., 2018).

In a rideshare system, a driver may travel longer than desired to pick up and drop off a passenger. Also, if there are other rideshare passengers in the vehicle at the time, their drop-off time may also be delayed. These inconveniences can be a detracting factor in the adoption of rideshare. To alleviate this, an incentive system can be established where the system, perhaps subsidized by a public agency, will pay some of the costs of the drivers and passengers for the inconvenience caused. This will encourage more people to participate in the rideshare system. In exchange, the city will get reduced traffic congestion and pollutant emissions.

In this study, we propose an online rideshare framework that can match passengers with drivers and provide efficient routes to drivers in real time. Our proposed framework is scalable to a city-scale network. We propose a mechanism for meeting point selection to minimize excessive driver detours. We also propose an online incentive and shared cost mechanism that can compensate drivers and passengers for inconveniences from a limited budget and share the cost of a ride fairly among passengers. Finally, we propose a machine learning-based prediction mechanism that can quickly predict future requests and redirect drivers to the future request location in order to increase passenger service rate. We utilize simulation to get updated travel times based on the traffic conditions on the road and also to study the effect ridesharing has on a city's traffic congestion.

The rest of the report is organized as follows. In the Literature Review section, we discuss some relevant literature, research gaps, and our contributions. In the Problem Description section, we describe our model mathematically and define our objective and the constraints. In the Proposed Framework section, we discuss our proposed methodology to solve the online ridesharing problem. In the Numerical Experiments section, we introduce our two datasets and the results and provide some insights. We end the report with Conclusions.

Literature Review

In this section, we review recent literature on dynamic online rideshare routing. We also discuss the literature on rideshare prediction models, incentive systems, and cost-sharing mechanisms. We also discuss the existing research gaps in the literature and the contributions of this study. We start the literature review by discussing online or dynamic routing algorithms.

Dynamic Vehicle Routing

Dynamic vehicle routing (DVRP) is a system where vehicles have to be routed to demand points that are not known in advance. Demands arrive dynamically during the planning horizon, and the routing system has to reroute the vehicles to accommodate these new demands. Various exact and heuristic methods have been proposed to solve the dynamic vehicle routing problems. Since the execution time budget to solve the routing problems is very small, most literature proposes heuristics and metaheuristics to solve them as quickly as possible. These heuristics achieve speedy execution in exchange for loss in solution quality. We now discuss some of the literature in this area.

One heuristic that has been used with great success is the insertion heuristic. The idea behind the insertion heuristic is that once a new demand arrives, the mechanism attempts to insert the request into the route in the most economical way. Ulmer et al. (2021) studied a dynamic vehicle routing problem (DVRP) with stochastic request arrival. They used an insertion heuristic, which incorporated an incoming request into the route or postponed the request to the next time step. In a previous paper, the same authors studied a same-day meal delivery problem with dynamically arriving orders and random ready times (M. Ulmer, 2017). Meal delivery apps such as DoorDash or Uber Eats are popular, and this is an application area for DVRP. They used an insertion heuristic here as well. Fikar (2018) studied a grocery delivery problem where the authors used an insertion-based heuristic to schedule the pickups and deliveries.

Metaheuristics are also widely used in solving DVRPs, such as Tabu Search, Genetic Algorithm, and Adaptive Large Neighborhood Search (ALNS). Tabu Search is a metaheuristic where an initial solution is modified until the best solution is found. It diversifies the search by prohibiting some modifications. Ferrucci and Bock (2015) used Tabu Search to solve a DVRP with en-route detours. In their problem, a driver can be rerouted during their trip to a customer's location. As Tabu Search is fast, they solved multiple instances of static problems in parallel. Genetic Algorithm is another metaheuristic widely used in the literature. The idea behind the algorithm is to mix multiple good solutions to produce a better solution. Novaes et al. (2015) studied a problem where auxiliary vehicles could be sent to a customer's location if the planned vehicle would not reach the customer in time. They used a genetic algorithm to solve the problem.

Although not common, some literature has proposed exact methods to solve the DVRP. Since solving a routing problem is NP-hard, these exact methods can solve only small- to medium-sized problems. These methods usually also use re-optimization. This means whenever there is a change in the demand, the problem is solved again from the beginning. Monroy-Licht et al. (2017) proposed a mixed integer linear program to solve the dynamic routing problem, which was solved from scratch with the newest information every time there was a failure to meet a new request. Amrouss et al. (2017) proposed an exact model for solving log-truck scheduling. The model was reoptimized anytime there was a change in the road network. For an extensive review of DVRP, we refer the readers to the review paper of Ojeda Rios et al. (2021). We now review some literature on rideshare routing and online rideshare routing.

Dynamic Rideshare Routing Problem

The rideshare routing problem has a limited set of drivers who have their own origin and destination instead of returning to the depot or some central location, as in the case of vehicle routing problems. Another difference is that, unlike VRPs, the drivers would like to reach their destinations with minimal delay. Therefore, they are not available throughout the planning horizon. These conditions make rideshare routing a more challenging problem to solve than the DVRP.

Dynamic rideshare routing is a version of the rideshare routing problem where the passenger and drivers dynamically enter or exit the system. As such, the routing system has to react in real time to the changing number of drivers and passengers. Although some literature has been published on dynamic rideshare routing, they mainly focus on taxi routing or ride-hailing routing (Agatz et al., 2011; Bertsimas et al., 2019; Ma et al., 2015; Simonetto et al., 2019; T. Wang et al., 2023). Taxi routing is similar to the Pickup and Delivery Problem, where a potentially unrestricted number of vehicles are available to serve the passengers' requests, and the drivers do not have their own specific origin and destination. Bertsimas et al. (2019) developed a heuristic to solve the taxi routing problem in New York City. Although their approach used a mixed integer model, by clever preprocessing and sparsifying the potential matches, they solved a large problem within minutes. Alonso-Mora et al. (2017) proposed a tripartite graph-based ride-hailing driver-passenger matching problem. They first constructed a graph of passengers who can share a ride. Then, they constructed another graph of those shared trips and drivers. Their matching algorithm then matched the drivers with the trips in the most optimum way. Xu et al. (2020) proposed a multi-hop ridesharing system where a passenger can be matched with multiple drivers along their route. This allowed for more flexibility and more ride availability. They proposed two efficient algorithms to solve this problem.

Although most of the literature studied a myopic problem, some literature proposes a non-myopic approach. In non-myopic approaches, future requests are predicted, and routes are modified to accommodate those future requests. Lowalekar et al. (2021) studied such a non-myopic approach for ride-hailing problems. They constructed all possible paths through the road network offline. During the planning horizon, these offline paths were matched with requests and drivers by solving a mixed integer program.

The literature discussed above all studies either ride-hailing or taxi routing problems. As mentioned before, these problems do not have driver origin-destination (OD) pairs, and drivers are usually available during the studied horizon to serve the passengers. The problem we study in this report is a dynamic ridesharing routing problem. Therefore, a different routing approach is necessary, which can solve this complicated problem with a minimal execution time budget and can scale to a city-scale network. We propose a greedy insertion-based heuristic to solve the large-scale dynamic rideshare routing problem. Next, we review some literature on incentives and shared cost mechanisms.

Incentive and Shared Cost Mechanisms for Rideshare Systems

In a rideshare system, drivers have to reach their destination within their time limits. As such, many drivers would be disinterested in participating in a rideshare system to avoid long ride times. Passengers would also like to reach their destination as quickly as possible. If a vehicle has to detour to pick up additional passengers, this may delay the drop-off of the passengers currently in the vehicle. This may detract some passengers from using the rideshare system. Providing incentives can be a way to solve this issue. Incentivizing people to participate in the rideshare system will benefit a city by reducing traffic congestion, pollutant emissions, and inaccessibility to transportation. Tafreshian and Masoud (2022) developed an incentive program that encouraged participation in the rideshare program. They provided behavioral incentives to drivers to extend their maximum ride time limits. Their proposed scheme is individually rational and budget-balanced. They studied a static system where the drivers and the passengers were known a priori, and incentives could be distributed equitably. Kumar et al. (2023) proposed an incentive system for economically disadvantaged drivers and passengers to promote fairness across various metrics. In this report, we study a dynamic rideshare system. Thus, we have to distribute the incentives from a limited monetary budget without knowing the future requests. We develop a system based on an online knapsack problem.

Another important aspect of the rideshare system is to share the costs of the drivers among the passengers in a fair manner. Various cost-sharing mechanisms have been proposed in the literature. Cipolina-Kun (2023) used game theory and reinforcement learning to form coalitions among passengers to share incentives and share costs that satisfy individual rationality. Furuhata et al. (2014) proposed a proportional online cost-sharing mechanism for a demand-responsive transit (DRT) system. They also put forth five desirable properties a cost-sharing mechanism should have. These are immediate response, individual rationality, budget balance, online fairness, and ex-post incentive compatibility. Hu et al. (2021) proposed a proportional online cost-sharing mechanism for ridesharing that satisfied these five properties and two additional properties. These were reduced burden for the first passenger and fairness in cost sharing. We use their cost-sharing mechanism to share the travel costs of the driver among the passengers. The next subsection reviews traveler request forecasting algorithms and simulation mechanisms.

Future Request Prediction and Simulation Mechanisms

Some literature has proposed machine learning-based methods to predict future requests. This prediction mechanism can be used in conjunction with a driver detour mechanism. In this case, if the likelihood of a future request is high, we can redirect a nearby driver to that location in anticipation of the request. This can increase the service rate of a rideshare system. To achieve this, an accurate and efficient request prediction mechanism is necessary. One way future requests can be predicted is by predicting a new graph of OD pairs for a future time period. Based on the historical OD graphs, Graph Neural Networks can predict where future requests will originate and where they will end. Wang et al. (2022) proposed a framework called 'Gallat', which used a graph attention network to predict passenger mobility. Shen et al. (2022) proposed a baseline gated attention neural network to predict the OD pairs. Their results showed that their model provided more accurate predictions than the Gallat Model. Cui et al. (2020) proposed a graph convolutional neural network to predict a city's traffic. Although the primary purpose of their model was to predict traffic, it can also be used to predict OD pairs for a rideshare network. However, in all of the studies mentioned above, the primary focus was to provide the most accurate prediction, and the execution time was not a concern. In our study, we need to predict future requests within a limited time budget. Therefore, we use a simpler prediction mechanism that can still provide accurate predictions and is computationally fast.

We need to use the most updated travel times to make the rideshare routes practical. Traffic conditions are constantly changing, especially in large metropolitan areas. Also, adding a large number of rideshare vehicles to a road network can change the dynamics of a traffic system, which cannot be measured from a mapping or routing application such as Google Maps or Waze. Deriving these dynamics analytically is also not possible since an urban area can have tens of thousands of traffic signals, speed limit changes, or freeway ramps. One alternative is to use microscopic traffic simulation software such as VISUM or SUMO. Some studies in freight routing have successfully integrated simulation into the routing mechanism. Zhao et al. (2018) proposed a co-simulation optimization approach where the routing algorithm sent the routes to the simulation layer, which then, in turn, provided updated travel times to the routing algorithm. The routing algorithm then used the updated travel times to modify the routes and balance the load of the road network. This back-and-forth between the routing and simulation layers continued until no improvement in routing could be achieved. Chen et al. (2021) used a co-simulation optimization approach for freight routing, where they used simulation to get the best estimates of cost when non-internal combustion engine vehicles are incorporated into a fleet of diesel trucks. To the best of our knowledge, no ridesharing study has incorporated simulation into the routing framework. This simulation provides the most updated traffic and travel time data and enables us to study the effect of ridesharing in an actual road network.

In this study, our goal is to develop a framework for dynamic rideshare routing. We develop a greedy insertion algorithm for large-scale and real-time routing. We incorporate a meeting point selection mechanism that can quickly find the optimum meeting point for a passenger, resulting in more travel cost savings. We use a prediction-based driver detour mechanism that can redirect drivers in anticipation of future requests. We propose an online incentive and a shared cost mechanism. Additionally, we incorporate a simulation mechanism that provides us with the most accurate estimates of travel times in a road network. In the next section, we describe the problem mathematically.

Problem Description

In this section, we mathematically describe the problem we are studying. As mentioned before, we are developing a system in which passengers' ride requests are dynamically matched to the drivers. Each passenger has an origin and a destination, a pickup time limit within which they must be picked up, and a maximum ride time limit within which they should be dropped off. In our problem, we assume pickup time window constraints are hard constraints and maximum ride time limit constraints are soft constraints. The maximum ride time limit constraints can be violated, albeit with a penalty. This may happen when adding another passenger to the route, which may delay their drop-off time, but pickup times still remain within their windows. However, in that case, we need to pay the passengers some incentives as compensation for their inconvenience. Drivers will have an origin and destination, a journey start time, and a maximum ride time limit. Like passengers, a driver's ride time limit is a soft constraint that can be violated. However, we also need to pay the drivers an incentive proportional to their ride time limit violation as compensation for their inconvenience. We assume a limited incentive budget, B. The system's objective is to minimize the total distance traveled by all the drivers and the incentive payments to the drivers and passengers while serving as many passengers as possible. We also consider flexible meeting points for the passengers where they may have to walk to their pickup point or from their drop-off point if it minimizes the driver's detour. We represent the road network as a collection of nodes N and edges $\mathcal{A} = \{(i,j): i,j \in N\}$. The nodes represent the junctions, and the edges represent the road segments. Each edge has an associated distance $d_{i,j}$ and a travel time $t_{i,j}$. We note that the travel times are the estimated travel times from the simulation framework that are updated at each planning epoch. The updating procedure and the simulation framework will be described in the next section.

We have a set of n drivers, which we denote using the set $V = \{v_1, v_2, \ldots, v_n\}$ and m passengers, which we denote using the set $P = \{p_1, p_2, \ldots, p_m\}$. We assume that we do not know their information beforehand and their information is only known to us when they enter the system. Each driver v and their relevant information is represented as $\{O_v, D_v, a_v, L_v, I_v, q_v, r_v, cl_v, s_v, R_v\}$. O_v and D_v represents the origin and destination of the driver respectively. a_v is the start time of their journey. L_v is the maximum ride time limit. We assume all drivers allow a certain extension to their direct travel time. Therefore, L_v is calculated as $(1 + ex_v)t_{O_v,D_v}$ where ex_v is the extension factor that the driver v allows and $t_{O_v,D_v}$is the estimated direct travel time between their origin and destination when a rideshare request is made. We assume the ride time limit constraint is a soft constraint, and the system pays an incentive to the driver proportional to the ride time limit violation, which we represent with I_v. q_v is the capacity of the driver's vehicle. r_v is the route assigned to the driver. Initially, when the driver enters the system, their route is just their origin and destination pair, i.e., $r_v = [O_v, D_v]$. As passengers are matched to the driver, the route is updated by the passenger’s pickup and drop-off points. cl_v represents the driver's current location in the road network. Finally, s_v is the driver's status, which is represented by a number from the set $\{0,1,2\}$. If it is 0, that means the driver hasn't entered the system yet. $s_v = 1$ means the driver is in the system currently, and we can assign passengers to their route. And $s_v = 2$ means the driver has reached their destination and exited the system after having entered the system. In this case, we cannot assign any more passengers to them. R_v is the set representing the passengers currently in the vehicle of driver v. In other words, these are the passengers who have been picked up but not dropped off.

Similarly, the information about passenger p is represented by $\{O_p, D_p, a_p, b_p, L_p, I_p, v_p, s_p, cs_p, W_p\}$ where O_p and D_p are the passenger's origin and destination. a_p is the earliest pickup time, whereas b_p is the latest pickup time. A passenger must picked up from their origin within this period. For this study, we assume that a passenger's pickup time window starts when the passenger requests a ride. Therefore, the earliest pickup time is also the passenger's request arrival time. Similar to a driver, L_p represents the maximum ride time for the passenger, and it is calculated similarly to that of a driver. I_p is the amount of incentive paid to the passenger for violating their desired destination time, which is proportional to their ride time violation. v_p is the driver matched to the passenger's request. s_p is the status of the passenger request. If $s_p = 0$ then it means the system has received the passenger's ride request, and the passenger is yet to be assigned to a driver. $s_p = 1$ means the passenger has been matched to a driver, whereas $s_p = 2$ means the passenger has reached their destination and exited the system after entering it. $s_p = 3$ means no match has been found for the passenger request, and the request has been rejected. In our problem, we keep the passenger in the system until their latest pickup time has passed. At that point, we reject the passenger's request and set s_p to 3. cs_p is the shared cost of the passenger. A passenger shares some of the driver's travel costs. We use a proportional cost-sharing mechanism described later in the report. Finally, W_p represents the maximum walking distance limit of passenger p.

We define the decision variable $\tau_{i,v}$ as the time when the driver v visits node $i \in N$. Our proposed mechanism tries to insert each incoming request p into the drivers' routes in a way that minimizes the weighted objective that considers the sum of the distance traveled by the drivers and the incentives paid to the drivers and the passengers.

When inserting a passenger request into a driver's route, the following constraints must be satisfied.

Each passenger must be picked up within their pickup time limits.

$$a_p \le \tau_{O_p,v_p} \le b_p \ \forall p \in P, \forall v \in V$$

Incentives for passengers are calculated as follows. Here, f_1 is the incentive payment per unit of passenger's ride time violation.

$$I_p = f_1 * \max\left(0, \left(\tau_{D_p,v_p} - \tau_{O_p,v_p} - L_p\right)\right) \forall p \in P$$

Incentives for drivers are calculated similarly. Here, f_2 is the incentive payment per unit of driver's ride time violation.

$$I_v = f_2 * \max\left(0, \left(\tau_{D_v,v} - \tau_{O_v,v} - L_v\right)\right) \forall v \in V$$

The total incentives paid to the drivers and passengers must be within the budget.

$$\sum_{p \in P} I_p + \sum_{v \in V} I_v \leq \mathrm{B}$$

The number of passengers in the vehicle at a time must not exceed the vehicle's capacity.

$$|R_v| \leq q_v \; \forall v \in V$$

In the next section, we present our proposed framework.

Proposed Framework

We define the planning horizon as T. We divide the planning horizon into epochs of size Δ. We assume that $\mathcal{T}$ represents the current time of the system. At each epoch, we process all the requests that arrived in the previous epoch. In other words, we process all the ride requests that arrive during $[\mathcal{T} - \Delta, \mathcal{T}]$. We also collect the information of any driver who entered the rideshare system during that period. For each request, we find the nearest k-drivers and try to match the request to the driver, which results in the least amount of detour for the drivers and the least amount of incentives paid to the driver and the passengers. We determine the best route for the matched driver to include the newly matched passenger. After finding the best match, we determine the meeting point for the passenger if it reduces the driver's detours. At each epoch, we forecast any future requests and redirect any driver that is nearby to the future request location. We also perform a simulation run to get the best estimate of the travel times. These updated travel times are used in the next epoch. Our proposed framework for a single epoch is summarized in Figure 1. Next, we describe each of the components of our proposed framework in detail.

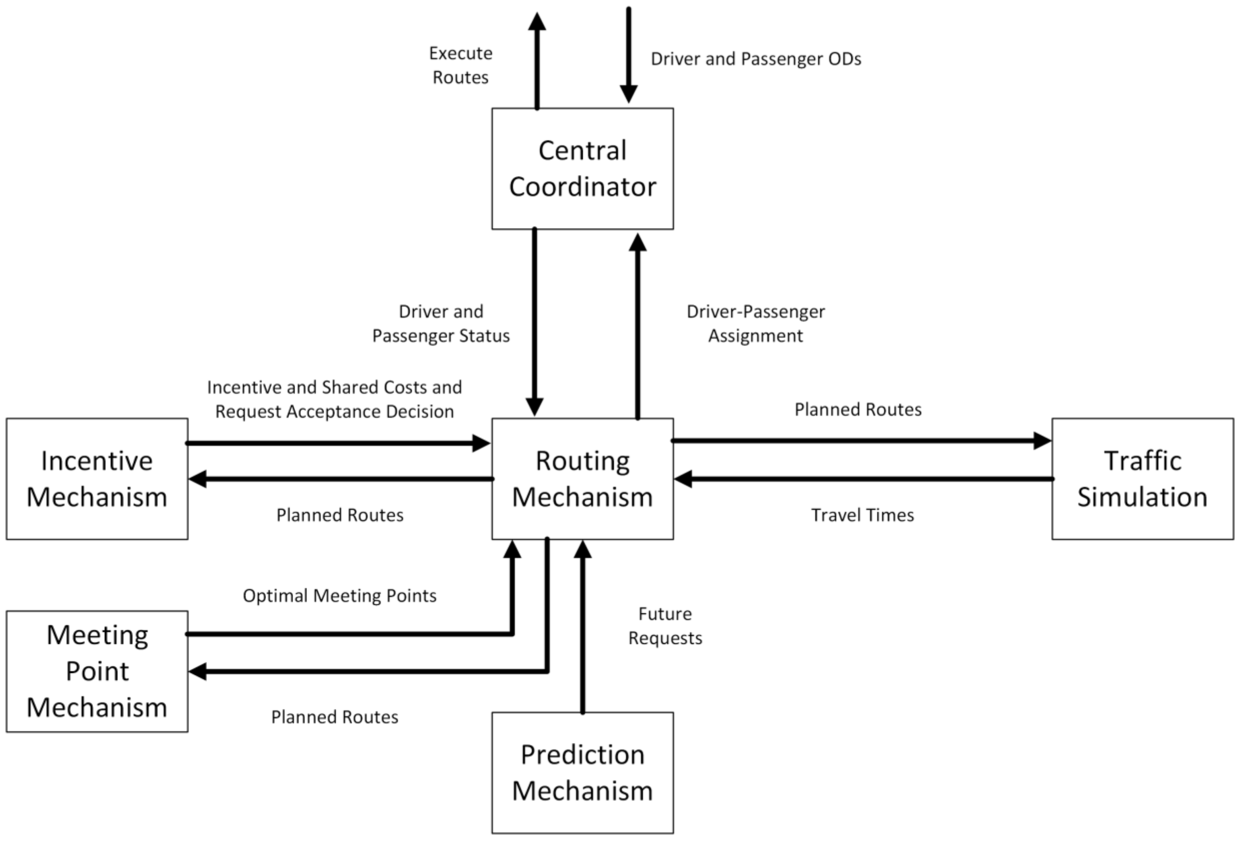


Figure 1. Proposed Framework

Central Coordinator

At the start of each epoch, the central coordinator collects all the information about the passenger requests and the drivers and sends all the information to the routing mechanism. After the routing mechanism has finished matching all the requests to the drivers and updated the driver routes, it sends this information back to the central coordinator. The central coordinator then sends the updated routes to the drivers. These routes are then executed during the following time epoch ($[\mathcal{T}, \mathcal{T} + \Delta]$). The central coordinator also informs the passengers about the status of their request. If the request has been accepted and matched to a driver, the central coordinator informs the passenger about the driver's information and the shared cost the passenger pays. If any incentives need to be paid to the drivers and any passenger in that driver's vehicle due to the inconvenience caused by adding the new request to the driver, the central coordinator informs them about it. The central coordinator acts as an interface between the system and the drivers and passengers. In this study, we assume that the drivers follow the routes that the system provides. We also assume that passengers will not cancel their rides after they have been matched with a driver.

Routing Mechanism

The routing mechanism receives all the information about the drivers and passengers' requests from the central coordinator. The routing mechanism also receives the updated travel times from the simulation module. Using this information, the routing mechanism attempts to match the passenger requests with the drivers in the most optimum way possible and also updates the drivers' routes. Since we are studying a city-scale network, we have to deal with potentially up to 10,000 requests an hour. We must also provide the matchings and routes before the next planning epoch begins. Hence, our routing mechanism has an execution time budget of at most Δ, which is a few minutes for practical purposes. Therefore, the routing mechanism needs to match and route up to a thousand requests within a few minutes. Thus, we need to use a fast heuristic for routing. For this reason, we developed a greedy insertion heuristic for the routing mechanism. As we later show in the numerical experiments, this heuristic can match hundreds of requests within seconds while achieving a good solution quality. To make the routing procedure even faster, for each request, we only consider k-drivers that are nearby both spatially and temporally. First, we describe our clustering procedure. Then, we describe our greedy insertion heuristic.

Request Processing and Clustering Mechanism

After the routing mechanism has received all the incoming ride requests, it sorts the requests based on the request arrival time. If there are j requests $p_1, p_2, \dots, p_j$ such that $a_{p_{(1)}} \leq a_{p_{(2)}} \leq \cdots \leq a_{p_{(j)}}$ then the routing mechanism sequentially processes the requests in order $p_{(1)}, p_{(2)}, \dots, p_{(j)}$. Whether a request is accepted by the platform depends on the shared cost of other passengers and the incentive budget. If including a passenger in a route increases the shared cost of the other passengers, or the incentive budget is exceeded by serving the request, it is rejected. This acceptance criteria is discussed later.

For each of the requests in this time epoch, we identify all the drivers that are currently in the system and select the drivers who can reach the passenger's origin before the passenger pickup time limit is exceeded. That is, we select the driver v for passenger p if,

$$\mathcal{T} + t_{cl_v,O_p} \leq b_p$$

Here, $\mathcal{T}$ is the current time of the system and t_{cl_v,O_p} is the travel time between the driver's current location in the road network cl_v and the passenger's origin O_p. If there are more than k drivers selected, we sort the drivers based on the distance between their current location and the passenger's origin, d_{cl_v,O_p} in a nondecreasing order and select the first k. For each request, this process has a worst-case run time of $O(n + nlog(n))$ where n is the number of drivers in the system.

Greedy Insertion Heuristic

Since we are processing the request in order of their arrival time, it is intuitive that the last position in a driver's route may be the best location to insert a new passenger request. This is because, at the last position of the route, the vehicle is empty (all the passengers have already been dropped off), and the only location that is in the driver's route is its destination. Therefore, in the last location, no incentives need to be paid to other passengers for their ride time limit violation. This is the idea behind our proposed greedy insertion heuristic. Given a passenger request and k-nearest drivers, our proposed heuristic attempts to find the best position in each of the driver's routes starting from the end. For each position, we calculate the cost of insertion. This cost is made up of two components. One is the travel distance, and the other is the incentives to be paid for the inconvenience to the driver and other passengers. If a worse position is found, we stop the search for a feasible location in that driver's route and move on to the next driver's route. After we have found all the feasible locations, we select the driver with the lowest insertion cost and insert the passenger's origin and destination. We now expand on the insertion cost calculation and our greedy insertion heuristic.

Let us say we have an incoming passenger ride request p with origin O_p and destination D_p. From the clustering mechanism, we have identified the k- closest drivers $v_1, v_2, \dots, v_k$. We start with driver v_1. Let us say the driver's route is $r_{v_1} = [O_{v_1}, Z_1, Z_2, \dots, Z_l, Z_{l+1}, Z_{l+2}, \dots, Z_{l+h}, D_{v_1}]$ where O_{v_1} and D_{v_1} are the driver's origin and destination and $Z_1, Z_2, \dots, Z_l, Z_{l+1}, Z_{l+2}, \dots, Z_{l+h}$ are the other passenger's origin and destination nodes that have already been assigned to the driver. We start by trying to insert O_p in the $(l + h)th$ position. To show how our heuristic calculates the costs, let us assume we want to insert O_p in the $(l + 1)th$ position. We find the cost of insertion by first calculating the cost of the distance increment $c_{\Delta d}$ due to inserting O_p in the $(l+1)\ th$ position. It is calculated as follows:

$$c_{\Delta d} = f_3 * (d_{Z_l,O_p} + d_{O_p,Z_{l+1}} - d_{Z_l,Z_{l+1}})$$

Here, f_3 is the cost penalty per unit of the distance increment. Next, we need to calculate the incentives that need to be paid to the drivers and the passengers who are dropped off after the $(l+1)$th position. To achieve this, we first calculate the time increment Δt. This is calculated as:

$$\Delta t = t_{Z_l,O_p} + t_{O_p,Z_{l+1}} - t_{Z_l,Z_{l+1}}$$

Then, the incentives for the passengers whose drop-off is between the $(l+1)th$ and $(l+h)th$ position are calculated as follows:

$$I_p = f_1 * \max\left(0, \left(\tau_{O_p,v} + \Delta t - \tau_{D_p,v} - L_p\right)\right)$$

The incentives for the driver are calculated as follows:

$$I_v = f_2 * \max\left(0, \left(\tau_{O_v,v} + \Delta t - \tau_{D_v,v} - L_v\right)\right)$$

The total cost of insertion of inserting into $(l+1)$th position is calculated as follows:

$$c_{\Delta d} + \left(I_v + \sum_{p \in R_v} I_p\right)$$

We start from $(l+h)th$ position and go backward until we have reached the driver's current location in the route. For each position, we calculate the insertion cost. We also check for the feasibility of the insertion. If inserting a new request violates other passengers' pickup time limits, then that position is infeasible. We continue as long as a position with better insertion cost is found. If not, we stop.

After we have found the best feasible location for O_p in driver v route r_v, we then find the best feasible position for D_p. The process is exactly the same. We start from the end of the route and stop when we have reached the insertion position of O_p restricting the drop-off to be after the pickup.

We save the best feasible position for both O_p and D_p and the associated insertion cost. After we have iterated through all the k -closest drivers, we select the driver v^* with the lowest insertion cost to be matched with the request p. We then insert the origin and destination in their respective best insertion position. Then, the updated route is sent to the meeting point selection module to select the optimum meeting points for the new passenger. The greedy insertion heuristic is summarized in Algorithm 1 (Figure 2).

```
Algorithm 1 Greedy Insertion
Require: List of $k$ nearby drivers, request $p$
 1: Initialize an empty feasible locations list
 2: for each driver in the nearby drivers list do
 3:     for each of the positions in the driver's route starting from the end of the
        route and ending at the driver's current location do
 4:         calculate the cost of insertion of the passenger's pickup point in each
            of the position
 5:         if  pickup time window feasibility is violated for other passengers then
 6:             break
 7:         end if
 8:         if a better feasible insertion position with lower cost found then
 9:             update the best feasible insertion position for the pickup point
10:         else
11:             break
12:         end if
13:     end for
14:     for each of the positions in the driver's route starting from the end of the
        route and ending at the best insertion position for the pickup point do
15:         calculate the cost of insertion in each of the positions for the passenger's
            drop-off point
16:         if pickup time window feasibility is violated for other passengers then
17:             break
18:         end if
19:         if a better insertion position with lower cost found then
20:             update the best feasible insertion position for the drop-off point
21:         end if
22:     end for
23:     Save the best feasible insertion position and associated cost for the pickup
        and drop-off in the feasible locations list
24: end for
25: Find the driver with the lowest total insertion cost from the feasible locations
    list
26: Insert the pickup and dropoff point into the best insertion position in the
    driver's route
```

Figure 2. Algorithm 1: Greedy Insertion

Meeting Point Selection Mechanism

The meeting point selection mechanism selects the meeting point for the newly inserted requests. Let the newly inserted request be p with an origin O_p and a destination D_p. Given the route, the meeting point selection mechanism outputs the new pickup point O_p' and drop-off point D_p' which are at most W_p apart from O_p and D_p respectively. Let, $rl_{O_p}^{x}$ and $rl_{O_p}^{y}$ be the x and y coordinates of O_p and A and B be the preceding node and succeeding node of O_p in the route. Then, the problem is equivalent to finding a point X in a circle centered around O_p with radius W_p such that the distance from A to X and then B will be the shortest. This point can easily be found by solving the following quartic equation.

$$4\upsilon(\upsilon\,\omega-\psi^2)x^4+4(\psi^2\,-\upsilon\,\omega)x^3+(\upsilon+2\psi\,+\omega-4\upsilon\,\omega)x^2+2(\omega-\psi)x+(\omega-1)=0$$

Finding the positive and real numbered root for this equation gives us the x coordinate of the desired point X. To find the y coordinate, we can substitute x into the following equation:

$$y=\frac{1+x-2\upsilon x^2}{1+2\psi x}$$

Where $\upsilon=\overrightarrow{O_pA}^T\,\overrightarrow{O_pA}$, $\psi=\overrightarrow{O_pA}^T\,\overrightarrow{O_pB}$ and $\omega=\overrightarrow{O_pB}^T\,\overrightarrow{O_pB}$. Here, $\overrightarrow{O_pA}$ and $\overrightarrow{O_pB}$ are $\mathbb{R}^2$ vectors representing x,y coordinates of points A and B considering O_p at $(0,0)$ of a $\mathbb{R}^2$ Euclidean plane. We take the positive and real number root of this equation and substitute them to the above equation to get the y coordinate value.

If there are multiple real positive roots of x then we take the y for which x and y are both positive. Then, the meeting point X is given by $X=x\overrightarrow{O_pA}+y\overrightarrow{O_pB}+\overrightarrow{O_p}$ where $\overrightarrow{O_p}$ is the true position of the center O_p. If X is not on a junction (node) of the network, we can quickly find the nearest junction to point X. Then the new junction is the new pickup point O_p' for passenger p. A closed-form solution exists to find the root of the quartic equation in $O(1)$ time. Therefore, the new meeting point can be found in $O(1)$ time. Given the preceding node and succeeding node, we can find the new drop-off point D_p' similarly. More details on the derivation of the quartic equation can be found in the study by Dessouky (2021). After the meeting points for the new request have been found, the routes are then sent to the incentives and cost-sharing mechanism. We describe this module next.

Incentive and Shared Cost Mechanism

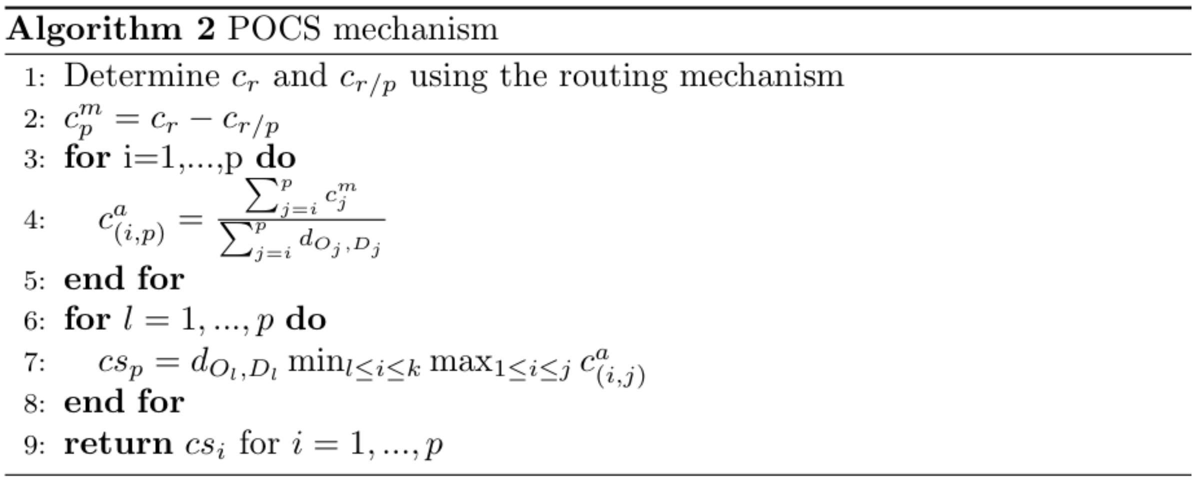
As mentioned before, this module calculates the shared costs the new passenger and any passengers currently assigned to the driver need to pay. It also calculates the incentives that need to be paid and, based on the remaining incentive budget, makes a request acceptance or rejection decision. We now describe the module.

As mentioned before, we have a limited budget B for the incentives. We have to pay incentives to drivers and passengers for the inconvenience caused by serving a new passenger from this budget without knowing how much incentives we have to pay for future requests. This problem is similar to the online knapsack problem, where the problem is to include the best set of items that maximize the value without exceeding the weight of the knapsack and not knowing the value of the future items. In our case, the capacity of the knapsack is our budget. The weights are the incentives we have to pay to serve a passenger, and the value is the benefit we get from serving a passenger. We assume that the benefit of serving a passenger is the cost savings of not taking a taxicab to the passenger's destination. In this case, we can use the acceptance criteria proposed by Chakrabarty et al. (2008). Our platform accepts a request if the following criterion is met.

$$\frac{\alpha_p}{\beta_p} \geq \Psi(\gamma_p)$$

Where $\Psi(\gamma) = \left(\frac{Ue}{L}\right)^{\gamma}\left(\frac{L}{e}\right)$. Here, α_p is the benefit of accepting a request p. β_p is the incentive to pay for accepting the request p. U and L are the upper and lower limits of the benefit-incentive ratio. e is the base of the natural log. And, γ is the fraction of the budget used so far. As long as U is not too high or L is not too low, the criterion makes sure the budget is not exceeded. Therefore, it is easy to show that our procedure satisfies the budget balance criteria. This algorithm has an approximation ratio of $\ln\left(\frac{U}{L}\right) + 1$.

For calculating the shared cost, we use the Proportional Online Cost Sharing (POCS) mechanism proposed by Hu et al. (2021). This mechanism satisfies the five desirable properties of cost-sharing outlined by Furuhata et al. (2014). This includes Immediate Response, Online Fairness, Individual Rationality, Budget Balance, and Ex-Post Incentive Compatibility. The shared cost mechanism is given in Algorithm 2 (Figure 3). The mechanism works by forming coalitions between the passengers with the same shared cost per demand (distance between origin and destination).ZStep 1 calculates the cost of the route with and without request p. c_r is the cost of the route r and $c_{r\setminus p}$ is the cost of the route without request p. c_p^m is, therefore, the marginal cost of inserting request p, which is calculated in step 2. $c_{(i,p)}^a$ is the coalition cost per demand value. Steps 3-5 calculates the shared cost per demand value for each of the passengers in the vehicle. Steps 6-8 of the algorithm forms the coalition among the passengers and calculate the shared cost for each of the passengers. For more details and proof of the properties, we refer to the papers by Furuhata et al. (2014) and Hu et al. (2021). After calculating the shared costs, we check if any of the passengers had their shared cost increased due to inserting passenger p. If this is the case, we reject the request p.



```
Algorithm 2 POCS mechanism
1: Determine $c_r$ and $c_{r/p}$ using the routing mechanism
2: $c_p^m = c_r - c_{r/p}$
3: for i=1,...,p do
4:     $c_{(i,p)}^a = \frac{\sum_{j=i}^{p} c_j^m}{\sum_{j=i}^{p} d_{O_j,D_j}}$
5: end for
6: for $l = 1, ..., p$ do
7:     $cs_p = d_{O_l,D_l} \min_{l \leq i \leq k} \max_{1 \leq i \leq j} c_{(i,j)}^a$
8: end for
9: return $cs_i$ for $i = 1, ..., p$
```

Figure 3. Algorithm 2: POCS Mechanism

We next describe our prediction mechanism.

Prediction Mechanism

One way we can increase the passenger service rate is by predicting where a request is likely to happen and redirecting any nearby drivers to that location in anticipation of the request. To achieve this, we collect historical data on past requests and train a machine learning model on this data to predict future requests. We construct a time series for each of the OD pairs in the network and train a LightGBM model to predict future requests. LightGBM is a fast version of the gradient-boosted regression trees, which forecast with limited execution time and good accuracy (Ke et al., 2017). This forecast determines how many requests will occur in each of the nodes in the next time epoch. To convert these numerical forecasts into likelihood, we assume the request arrival follows a Poisson distribution. We chose the Poisson distribution because it applicable for both low and high number of arrival requests. Then, analyzing the past data, we can determine the Poisson arrival rate λ. Let us assume that the forecasting model determines there will be F_i requests at node i in the next time epoch. Then, if we have seen F_i' requests so far at node i, we can determine the probability of seeing $F_i - F_i'$ requests using a Poisson CDF:

$$\mathrm{Pr} = \sum_{k=0}^{F_i - F_i'} \frac{\lambda^k e^{-\lambda}}{k!}$$

If Pr is greater than some predetermined threshold, we search for a nearby driver who does not have any passengers assigned to it and redirect the driver to the node. The node is inserted into the driver's route. If a request happens, then the driver will be matched to that request. If not, the platform compensates the driver for the detour if their ride time limit is exceeded.

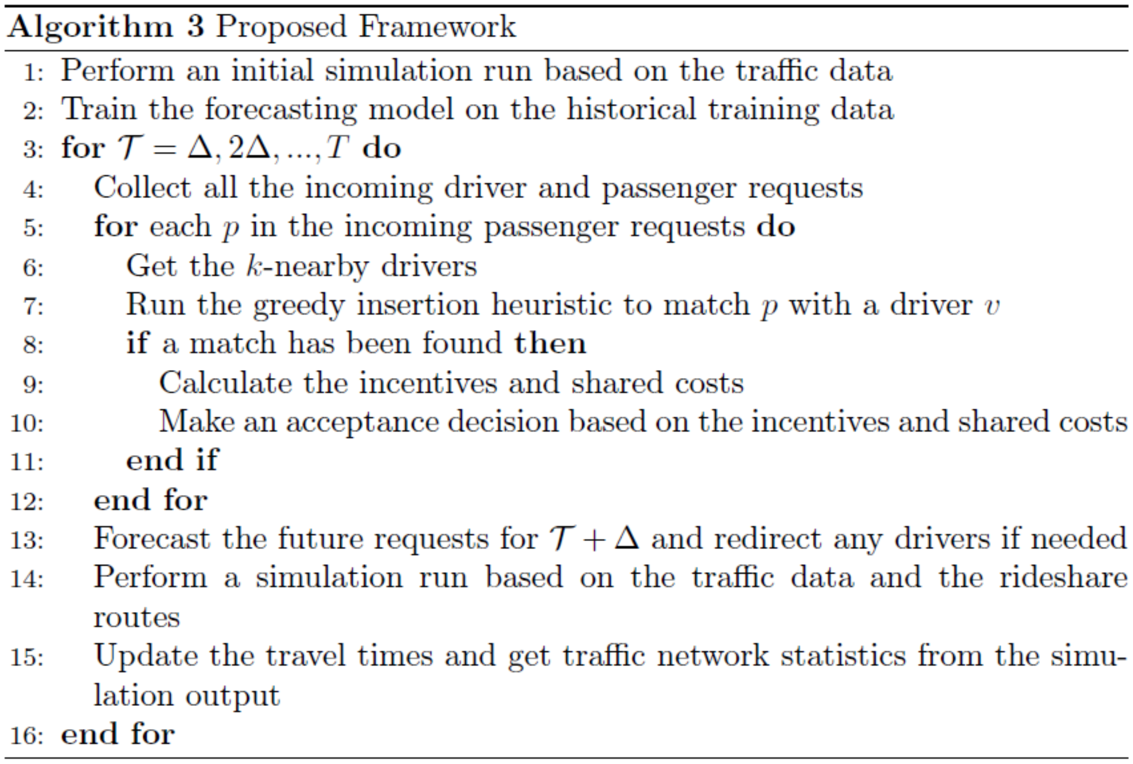
Simulation Mechanism

To make the driver's routes more effective, we need to use the updated travel times in the road network to get a better understanding of the traffic situation. This makes the planned routes more robust to the traffic condition. We can get the updated travel times via sensor data installed in the junctions. However, these data do not incorporate the effect the ridesharing routes have on the traffic conditions of the network. Also, without sensor data in every junction, we cannot estimate the traffic situation. In most US cities, junctions (traffic signals, freeway ramps) do not have sensors installed. In this case, traffic simulation can be useful in getting a better estimate of travel times. In our study, we use the open-source traffic simulator SUMO (Simulation of Urban Mobility) (Krajzewicz, 2010). Although some studies in the literature used other simulation software such as PTV VISIM and VISUM, we choose SUMO because it is open source and free to use and provides an easy-to-use API for Python which our other proposed mechanisms are developed, allowing a seamless connection with other modules. Given a road network and the number of vehicles on every edge, it provides the most updated travel times. Also, it allows for the simulation of custom routes. Therefore, we can use this traffic simulation software to simulate our ridesharing system and understand its effect on a city's traffic network.

Before the planning horizon begins, we perform a simulation run from the historical traffic data in the road network. If historical data is unavailable, we can assume a probability distribution of traffic throughout the network. This simulation run provides the initial estimates of the travel times. At the end of each epoch, we run the simulation again but with rideshare routes included. If a passenger is rejected, we assume the passenger takes a taxi or uses a personal vehicle. We simulate these trips as well as the rideshare trips. We update the travel times based on this simulation output, which will be used in the next epoch to match drivers with passengers and plan the routes. The SUMO simulation software can also output statistics about the road network. This includes the average duration of each route, the average time vehicles were stuck in traffic jams, and the time loss due to the high traffic volume in the network. Analyzing these outputs can demonstrate the benefits of the rideshare system in alleviating traffic jams in a city.

Overall Framework

We next summarize the overall framework. Algorithm 3 (Figure 4) shows the summary. Before the planning horizon begins, we perform a simulation run based on the historical traffic data or a probabilistic distribution of traffic if historical traffic data is unavailable. We also gather historical data on ride requests and train our forecasting model on the data. We do the training offline to save time during the planning horizon. After the planning horizon begins, at the start of each epoch, the central coordinator collects information about all the incoming ride requests and drivers that arrived during the previous epoch. This information is then sent to the routing mechanism. The routing mechanism uses the distance data and the most updated travel time data to match the ride requests with the drivers currently in the system. It modifies the routes of the drivers who were matched with a ride request using our proposed greedy insertion routing algorithm. These planned routes are then sent to the meeting point selection mechanism, which then selects the optimum meeting points for the matched requests and sends the information back to the routing mechanism. These routes and all relevant information are then sent to the incentive and shared cost mechanism. This mechanism calculates the shared costs and the incentives that need to be paid and makes an acceptance decision for the requests. After all the incoming requests have been processed, the prediction mechanism forecasts requests for the next epoch and redirects drivers to locations with a high probability of future requests. We also perform a simulation run of the traffic and the rideshare routes, and based on the output, we update the travel times. We then move on to the next planning epoch.



Algorithm 3 Proposed Framework

```
 1: Perform an initial simulation run based on the traffic data
 2: Train the forecasting model on the historical training data
 3: for $\mathcal{T} = \Delta, 2\Delta, ..., T$ do
 4:     Collect all the incoming driver and passenger requests
 5:     for each $p$ in the incoming passenger requests do
 6:         Get the $k$-nearby drivers
 7:         Run the greedy insertion heuristic to match $p$ with a driver $v$
 8:         if a match has been found then
 9:             Calculate the incentives and shared costs
10:             Make an acceptance decision based on the incentives and shared costs
11:         end if
12:     end for
13:     Forecast the future requests for $\mathcal{T} + \Delta$ and redirect any drivers if needed
14:     Perform a simulation run based on the traffic data and the rideshare
        routes
15:     Update the travel times and get traffic network statistics from the simu-
        lation output
16: end for
```

Figure 4. Algorithm 3: Proposed Framework

Numerical Experiments

In this section, we present the numerical experiments conducted. We perform numerical experiments on two datasets—one in an urban setting and the other in a rural setting. For the urban experiments, we chose the New York Taxicab and Limousine Commission dataset. For rural experiments, we chose the area of Tulare and Kern Counties, California. First, we describe the urban dataset, and then we examine the performance of our proposed methodologies. We evaluate how our proposed methodologies scale up to large-scale rideshare networks. We investigate how ridesharing can help alleviate a city's traffic congestion problem and how beneficial flexible meeting points are. Then, we describe the rural dataset and conduct the same set of experiments. All the experiments were conducted on an Intel Xeon computer with 32 GB of RAM. All the codes for the mechanism were written using Python 3.12.0. We also used the SUMO (Simulation of Urban Mobility) software as our simulation tool. We now describe our urban dataset.

Description of the Urban Dataset

For numerical experiments in the urban setting, we choose the New York Taxicab and Limousine Commission dataset (New York (N.Y.) Taxi And Limousine Commission, 2019). This dataset contains data about taxi trips in New York City from 2009 to the present. It contains data on over a billion trips of yellow, green, and black taxis and ride-hailing vehicles (Uber and Lyft). NYC government has divided New York City into 263 zones for data collection purposes. Each entry in the dataset contains the pickup zone number, the drop-off zone number, pickup time, trip duration, trip distance, fare paid, taxicab id, and other information. For our experiments, we chose the records of ride-hailing vehicles (defined in the dataset as For-Hire Vehicles). We focus on the trips from January 1st to December 31st, 2023. We use the data from January to November as training data for our machine learning model. For constructing test instances, we focus on the trips occurring in December 2023. We further focus only on trips starting and ending in Manhattan. The Manhattan borough is divided into 69 zones. A map of the zones provided by the NYC Taxi and Limousine Commission is shown in Figure 5. Among them, five zones are islands that are only reachable by waterways and no road connecting them. Hence, we exclude them from consideration. To construct instances, we randomly sample rows from the dataset with pickup times between 4 pm and 5 pm. We chose this time period because traffic conditions are at their peak between these hours. An hourly distribution of the number of vehicles throughout the day in Manhattan is shown in Figure 6. This Figure 6 was obtained by analyzing the data from traffic sensors installed throughout Manhattan. This data can be found at https://catalog.data.gov/dataset/traffic-volume-counts. We use this distribution data to simulate random traffic in the road network. We construct both driver and passenger information from the NYC taxicab dataset. We assign the pickup zones as the origins and the drop-off zones as destinations for both drivers and passengers. We assign the pickup times as the request time for passengers or journey start time for drivers. To construct the pickup time limit for passengers, we added 15 minutes to the request time.

We assume all drivers and passengers allow a 50% extension to their direct travel time. To get the distance matrix, we use OpenStreetMap router API. This API gives us the distance in kilometers. We also use OpenStreetMap to download the road network map for Manhattan, which we use for the SUMO simulation software.

For experimentation purposes, we assume our planning horizon is 1 hour. We set the planning epoch to be 5 minutes. That means every 5 minutes, our proposed framework processes all the request data and sends them to the routing mechanism. As mentioned before, a passenger's pickup time limit is 15 minutes after their request time. That means our framework needs to assign the passenger to a driver's route, and that driver must pick up that passenger within 15 minutes of the passenger's request time. We set the passengers' walking limit to 0.84 kilometers. That means a passenger is willing to walk a maximum of 0.84 kilometers to or from their meeting point from their originally requested origin or destination. For each passenger request, our routing mechanism considers the 20 closest drivers. The travel cost of a driver is $0.187/km, which includes fuel cost, maintenance cost, and vehicle depreciation cost. We set a driver's fuel cost to be $0.097 per kilometer. We got this number by assuming a vehicle's average gas mileage to be 20 miles/gallon or 32.19 km/gallon in the city and the average gas price of $3.16/gallon in New York City (https://gasprices.aaa.com/?state=NY). The average maintenance cost of the vehicle is assumed to be $0.06/km, and the average depreciation cost is assumed to be $0.03/km (https://newsroom.aaa.com/wp-content/uploads/2022/08/2022-YourDrivingCosts-FactSheet-7-1.pdf). This travel cost is used to calculate a passenger's shared cost. We set the hourly incentive budget to be $2,000. In a later subsection, we study how changing this budget will affect the performance of the rideshare system. The upper and lower limits of the benefit-incentives ratio are set to 10 and 1, respectively. For all the drivers and passengers, we assume the system will pay $1.00 per minute violation of their maximum ride time. The parameters and their values are summarized in Table 1.

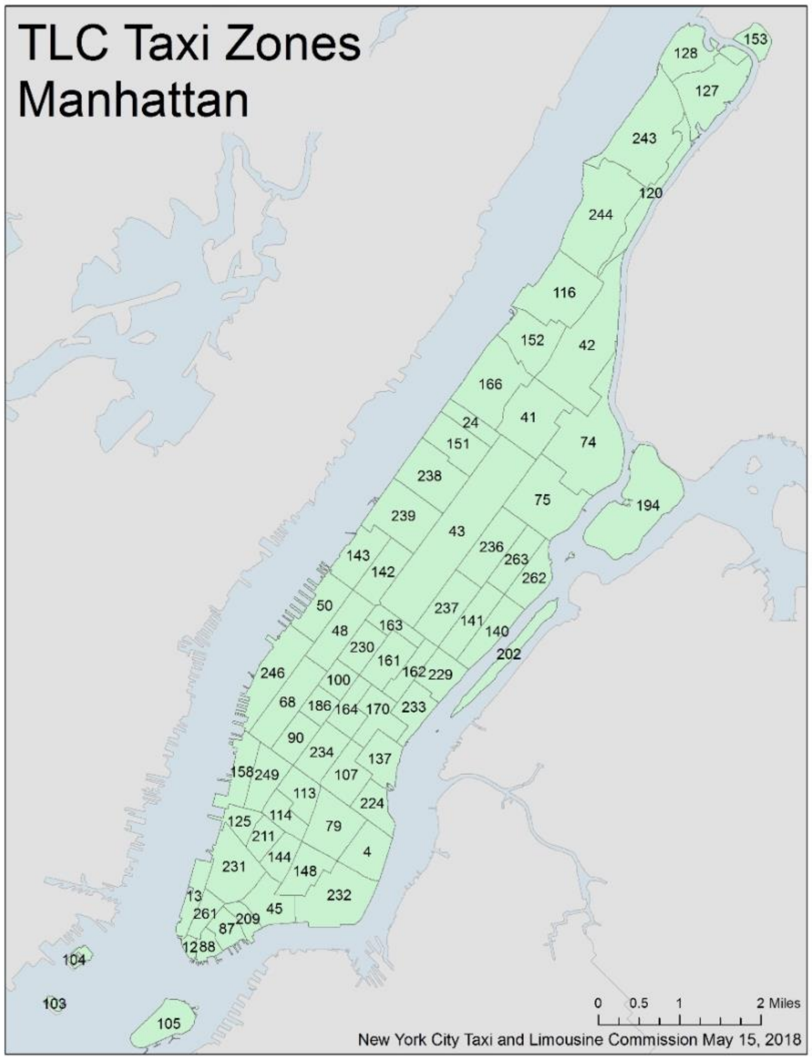


Figure 5. Taxi Zones in Manhattan; source: (New York (N.Y.) Taxi And Limousine Commission, 2019)

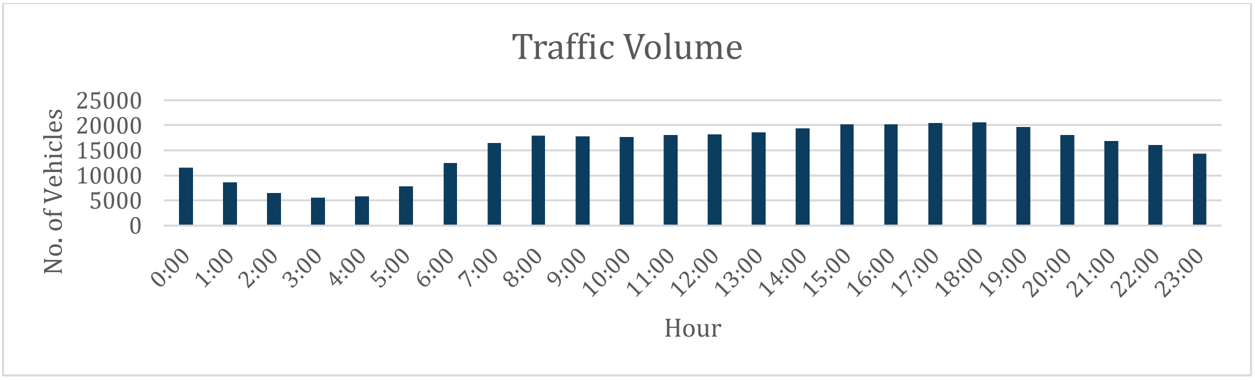


Figure 6. Traffic Volume Distribution in Manhattan throughout the Day

Table 1. Parameters and Their Values for the NYC Dataset

Parameter	Values
Planning Horizon, T	60 minutes
Planning time epoch, Δ	5 minutes
Pickup time limit, b_p	Pickup start time (request time), a_p+15 minutes
Hourly incentive budget, B	$2000
Travel cost per km	$0.187
Number of nearby drivers, k	20
Passenger walking limit, W_p	0.84 km
Driver ride time extension factor, ex_v	0.5
Passenger ride time extension factor, ex_p	0.5
Incentive payment per minute of maximum ride time violation, f_1 and f_2	$1.00
Penalty for per kilometer distance increment, f_3	$1.00
Upper limit of the benefit-incentive ratio, U	10
Lower limit of the benefit-incentive ratio, L	1

Next, we discuss the performance and scalability of our proposed framework on various-sized instances.

Performance of the Proposed Framework in the NYC Dataset

In this subsection, we report the performance of our proposed framework on instances of different sizes. The number of drivers in the rideshare system ranges from 1,000 to 5,000. The number of passengers range from 1,000 to 10,000.

We first study the effect ridesharing has on a city's road network by running our simulation software on two scenarios. In one scenario, we do not have any ridesharing. We assume everyone uses their own vehicles to reach their destination and drive solo. In this case, we simulate the routes consisting of the origin and destination pairs of all the passengers and drivers, as well as other traffic on the road network. In the second scenario, we simulate a ridesharing system where some of the passengers have been picked up or dropped off by rideshare drivers. In this case, we assume the passengers whose requests were not accepted will travel solo using a private vehicle to their destination. In this scenario, we simulate the rideshare route, unserved passengers OD routes, and other traffic in the road network. From the simulation output, we report the total vehicle distance traveled, average waiting time, and average time loss. The total vehicle distance traveled is defined as the distance traveled by the rideshare drivers and the passengers whose requests were not accepted who we assume to drive solo. The waiting time is the amount of time a vehicle is not moving. This may be due to traffic congestion, waiting at a traffic signal, or a stop or yield sign. The time loss includes the waiting time plus any additional time loss by driving at a speed less than the speed limit. We report these parameters for two scenarios and the improvement we get in these parameters by introducing a rideshare system to a city. The improvements are calculated using the following formula:

$$\frac{\text{(Output from solo driving scenario-Output from ridesharing scenario)}}{\text{Output from solo driving scenario}}$$

The results are shown in Table 2. From the Table 2, we can see that ridesharing can reduce the total vehicle distances traveled in a city by 13% on average across all test instances. This improvement decreases as the instances grow larger. This is due to the low service rates with respect to the number of passengers for larger instances, as can be seen in Table 3. Regarding average waiting time and time loss, we see ridesharing can reduce those by 4% and 3% on average, respectively. Therefore, ridesharing can also reduce time loss due to traffic congestion. We gain two insights from these results. First, rideshare can significantly reduce the total vehicle distance traveled, reducing traffic congestion and pollutant emissions. Second, increasing the rideshare service rate can further reduce the total vehicle distances traveled.

Table 2. Total Vehicle Distance Traveled, Waiting Time, and Time Loss across All Instances in Solo Driving vs. Ridesharing

		Solo Driving			Ridesharing			% Improvement		
No. Drivers	No. Passengers	Total Vehicle Distance Traveled (km)	Avg. Waiting Time (min.)	Avg. Time Loss (min.)	Total Vehicle Distance Traveled (km)	Avg. Waiting Time (min.)	Avg. Time Loss (min.)	Total Vehicle Distance Traveled	Avg. Waiting Time	Avg. Time Loss
1000	1000	17508.12	8.95	13.67	13093.00	8.94	13.65	25%	0%	0%
1000	2000	27471.58	9.03	13.80	22782.50	8.49	13.23	17%	6%	4%
2000	2000	36119.72	9.84	14.60	29875.36	8.88	13.65	17%	10%	6%
2000	4000	55753.34	9.48	14.34	49217.40	9.11	13.94	12%	4%	3%
3000	3000	55112.00	9.36	14.20	47641.44	8.98	13.73	14%	4%	3%
3000	6000	84087.10	9.29	14.27	76493.86	8.81	13.68	9%	5%	4%
4000	4000	73763.52	8.08	13.12	64550.22	7.95	13.11	12%	2%	0%
4000	8000	112017.70	9.21	14.18	103583.72	8.94	13.96	8%	3%	2%
5000	5000	93043.14	9.20	14.19	84526.00	8.72	13.69	9%	5%	4%
5000	10000	140571.71	9.55	14.60	132259.20	9.34	14.41	6%	2%	1%
Average		**69544.79**	**9.20**	**14.10**	**62402.27**	**8.82**	**13.71**	**13%**	**4%**	**3%**

Next, we report the passenger service rate, budget usage rate, average occupancy per vehicle, and the average trip distance of the rideshare vehicles. The results are shown in Table 3. From the Table 3, we can see the average passenger service rate is 34%. The budget usage rate is 100% across all instances. The hourly incentive budget stays at $2,000 across all instances. As the instances grow larger, there is not enough budget to provide incentive payment to drivers and passengers for their maximum ride time violation. Therefore, we cannot service more passengers without violating other drivers' and passengers' maximum ride time limits. As a result, the service rate decreases with increasing problem instance sizes. The average occupancy rate also decreases as the passenger service rate decreases. The average occupancy rate is 1.44 across all problem instances. The average distance traveled by a rideshare vehicle is 10.75 kilometers. In a later subsection, we will investigate how changing the incentive budget affects the rideshare service rate.

Table 3. Passenger Service Rate, Budget Usage Rate, Occupancy Rate, and Trip Distance per Vehicle Across All Instances

No. Drivers	No. Passengers	Passenger Service Rate (%)	Percent Budget Used	Avg. Occupancy per Vehicle	Trip Distance per Vehicle (km)
1000	1000	89%	100%	1.89	12.34
1000	2000	41%	100%	1.82	12.35
2000	2000	49%	100%	1.49	10.57
2000	4000	25%	100%	1.49	10.95
3000	3000	36%	100%	1.36	10.40
3000	6000	18%	100%	1.35	10.48
4000	4000	32%	100%	1.32	10.16
4000	8000	14%	100%	1.27	10.22
5000	5000	22%	100%	1.22	9.97
5000	10000	10%	100%	1.21	10.10
Average		**34%**	**100%**	**1.44**	**10.75**

Next, we report the average shared cost, the percentage of rideshare participants receiving incentives, and the average incentive payment across all instances in Table 4. The average shared cost was $1.68, which is significantly lower than what a ride-hailing or taxi ride would cost. Therefore, ridesharing can be an effective way to provide low-cost rides to commuters. The average percentage of participants receiving incentive payments is 20%. A decreasing pattern can be seen in the percentage of participants receiving incentive payments. As we have seen in Table 3, the budget usage rate is 100% across all instances, and the incentive budget is fixed at $2,000. Therefore, as the number of participants increases, the percentage of participants receiving incentive payments decreases. The average incentive payment is $2.94. From this discussion, we can conclude that rideshare can provide low-cost and flexible rides to passengers and alleviate a city's traffic congestion and pollution problem.

We next report the execution times in Table 5. In this Table 5, we report the average execution time of the proposed routing, meeting point section, and simulation mechanism per planning epoch. From the Table 5, we can see that our routing procedure is fast and scalable for a city-sized network. On average, the routing mechanism takes about 1.76 seconds per planning epoch of 5 minutes. Even for instances with 5,000 drivers and 10,000 passengers, our proposed greedy insertion heuristic takes about 4.00 seconds per planning epoch. Also, the meeting point selection procedure is fast, taking only 0.16 seconds on average. The only time-consuming step is the simulation. Each run of the simulation takes about 123.70 seconds on average. The maximum time it took was 206.38 seconds (3.43 minutes). From the results, we can conclude that our proposed framework can solve the rideshare routing and meeting point selection problem on a city-scale network within the planning epoch of 5 minutes.

Table 4. Average Shared Cost, Incentive Recipient Rate, and Incentive Payment Across All Instances

No. Drivers	No. Passengers	Avg. Shared Cost ($)	Percent Participating Receiving Incentive	Avg. Incentive Payment ($)
1000	1000	1.63	52%	3.11
1000	2000	1.64	31%	3.46
2000	2000	1.76	25%	2.95
2000	4000	1.66	18%	3.05
3000	3000	1.75	19%	2.79
3000	6000	1.65	13%	2.90
4000	4000	1.73	15%	2.69
4000	8000	1.66	16%	2.52
5000	5000	1.72	11%	2.78
5000	10000	1.64	7%	2.95
Average		**1.68**	**20%**	**2.94**

Table 5. Execution Time of Our Proposed Framework Across Different Problem Instances

No. Drivers	No. Passengers	Avg. Routing Exec. Time (s)	Avg. MPS exec. Time (s)	Avg. Sim. Exec. Time(s)
1000	1000	0.35	0.16	102.45
1000	2000	0.55	0.15	206.38
2000	2000	0.68	0.13	108.28
2000	4000	1.31	0.16	104.38
3000	3000	1.21	0.17	108.81
3000	6000	2.19	0.17	105.84
4000	4000	1.84	0.16	170.23
4000	8000	3.25	0.17	107.86
5000	5000	2.25	0.15	113.39
5000	10000	4.00	0.16	109.42
Average		**1.76**	**0.16**	**123.70**

Effect of Flexible Meeting Points

In this subsection, we study the effect of flexible meeting point selection. We compare the total travel distance of flexible meeting points to the total travel distance where we do not have any flexibility in meeting point selection. In the second case, the drivers have to go to the passenger's originally requested origin or pickup points. The results are shown in Table 6. From the results, we can see that incorporating meeting points and allowing passengers to walk up to 0.84 kilometers can reduce the total distance traveled by the rideshare drivers by 4% on average. Also, a decreasing trend in improvement can be seen as the instances get larger. This is because, for larger instances, the total distance traveled by all the drivers is much larger than the distance savings we get from passengers' walking. Also, for larger instances, the service rate is lower. This is reflected in the average walking distances per rideshare passenger, which also decreases as the problem instances grow larger. Because of these two reasons, we see a decreasing trend in distance savings. If we look at Table 5, we see that the meeting point selection mechanism only takes about 0.16 seconds per planning epoch (5 minutes in these experiments). Also, although we set the maximum walking distance limit to 0.84 km, on average, passengers have to walk only 0.39 kilometers to achieve the lowest driver detour. Therefore, incorporating flexible meeting points can be extremely beneficial in reducing driver detours and encouraging drivers to participate in the rideshare system.

Table 6. Total Travel Distance in Flexible Meeting Points vs. No Meeting Points

No. Drivers	No. Passengers	Total Travel Distance (km)			% Improvement
		Flexible Meeting Points	No Meeting Points	Avg. Walking Distance (km)	
1000	1000	12341.72	13609.88	0.48	9%
1000	2000	12346.81	13455.72	0.45	8%
2000	2000	21134.64	22251.43	0.40	5%
2000	4000	21904.89	23024.58	0.39	5%
3000	3000	31185.95	32396.23	0.39	4%
3000	6000	31448.43	32575.05	0.38	3%
4000	4000	40657.96	41510.26	0.33	2%
4000	8000	40871.18	41988.24	0.36	3%
5000	5000	49837.38	50909.44	0.35	2%
5000	10000	50494.10	51551.25	0.35	2%
Average		**31222.31**	**32327.21**	**0.39**	**4%**

Effect of Incentive Budget on the Performance of the Rideshare System

In this subsection, we study how changing the budget for incentives can affect the performance of the rideshare system. We take an instance with 5,000 drivers and 5,000 passenger requests in an hour, and we increase the budget from $1,000 to $10,000. We investigate the effect the incentives budget plays on the passenger service rate and the budget usage rate. Figure 7 shows the change in passenger service rate with the change in the incentives budget. As we can see from the Figure 7, when the incentive budget is low, the passenger service rate is also low. In these cases, we do not have enough budget to pay incentives to the drivers for their inconvenience. As a result, drivers are unwilling to take a detour to pick up or drop off other passengers, and fewer passengers are assigned to the drivers. As the budget increases, we see an increasing trend in the percentage of passengers served, which increases up to 59% with a $6,000 budget. From the figure, we can see increasing the budget beyond this does not change the passenger service rate. This is because of the passenger's pickup time window constraints. The remaining passengers cannot be served irrespective of the incentive budget. In this case, no driver is nearby to serve those requests within the passenger’s pickup time window. Since increasing the passenger service rate decreases the total vehicle miles traveled, as we have seen previously, increasing the incentive budget and encouraging participation in the rideshare system can be a great way to reduce the traffic congestion and pollution problem of densely populated urban areas like Manhattan. In our experiments, the optimum budget is $6,000 per hour.

Next, we look into the budget usage rate. This is shown in Figure 8. An inverse pattern of Figure 7 is seen in Figure 8. When the budget is low, the entire budget will be used to incentivize drivers and passengers. This stops at $5,000, beyond which the budget use decreases.

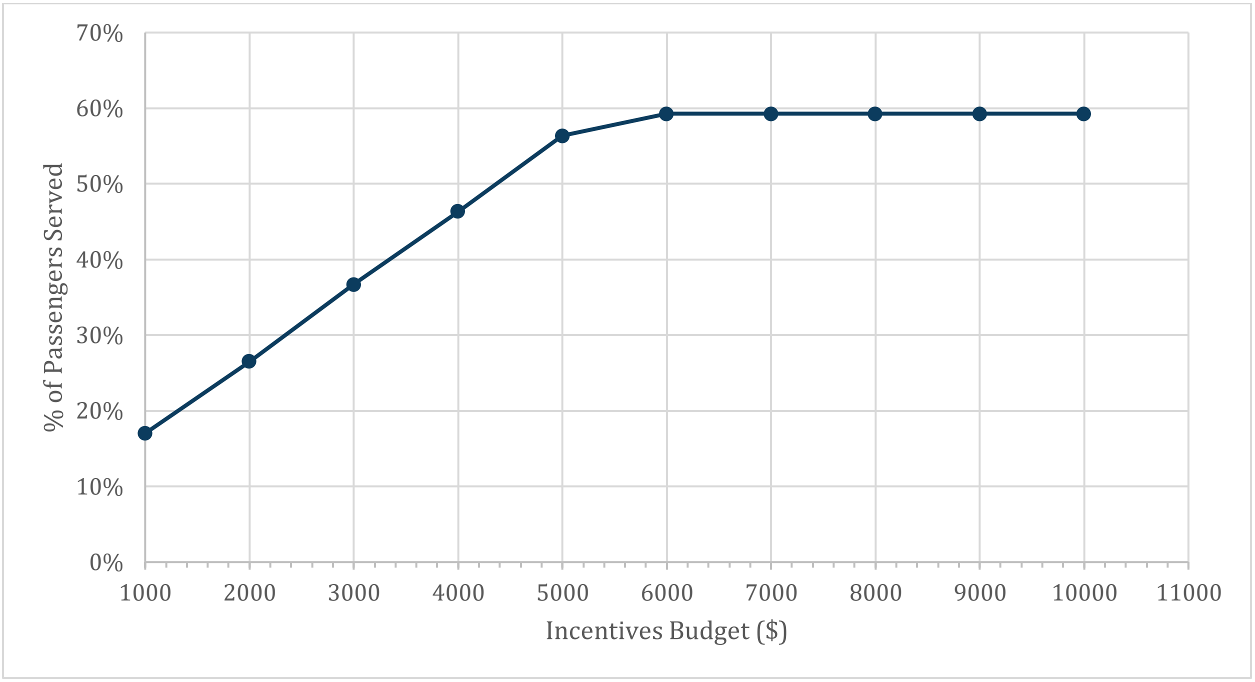


Figure 7. Change in the Percentage of Passengers Served with Change in Budget

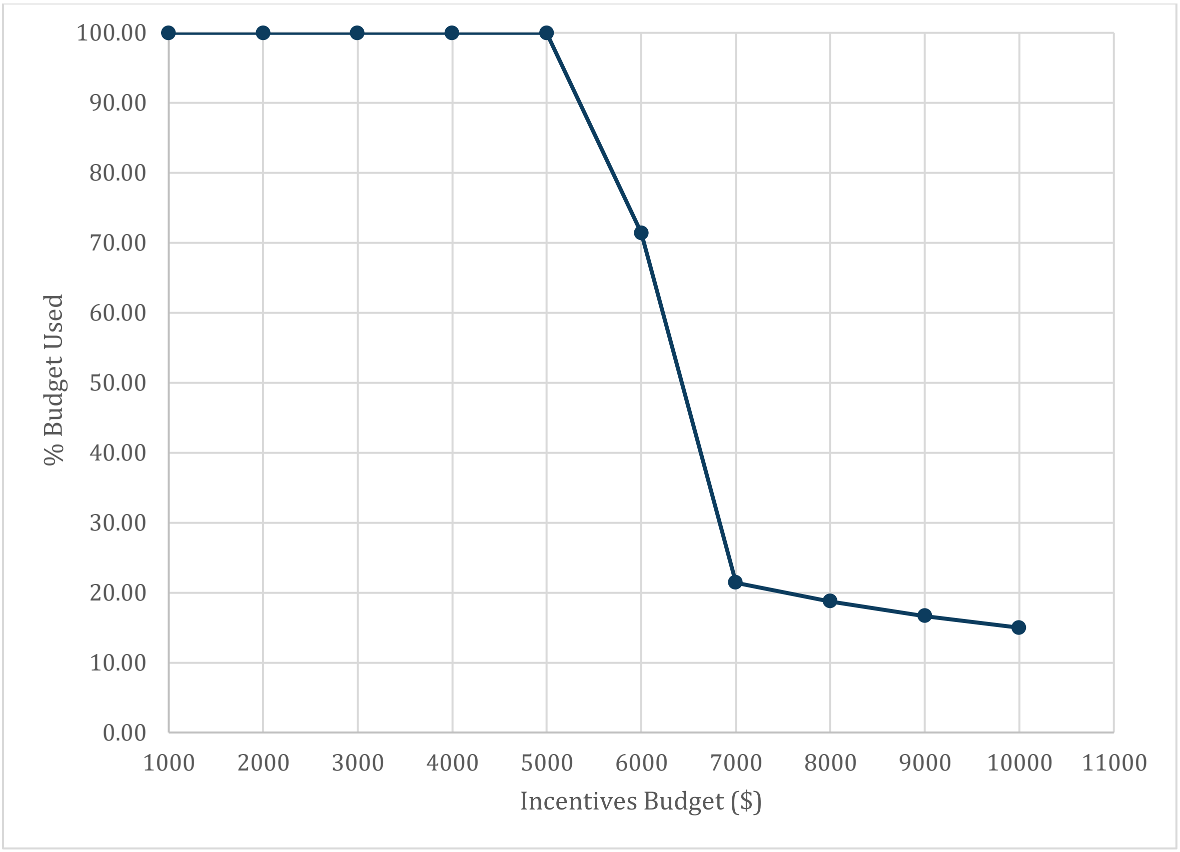


Figure 8. Change in the Percentage of Budget Used with Change in Budget

Next, we describe our rural area dataset and examine the performance of the rideshare network on this dataset.

Description of the Rural Dataset

We now show how rideshare systems perform in a rural area where the population is sparse, and public transport is not available. However, the rideshare literature does not offer any dataset in this regard. Therefore, we construct our own dataset. We chose the area centered around the town of Delano, California. The area sits between Kern County and Tulare County. Although there are many areas like this throughout the United States, we wanted to focus on California, especially the valley area, due to its geographic proximity. This area has two towns in the middle, Delano and McFarland. The area surrounding these towns is mostly farmlands with sparse population density. This area provides a perfect opportunity to study the rideshare system in the rural area. A map of the area is provided in Figure 9. The map is sourced from Google Maps.

We prepare the dataset in a similar way to the New York Taxicab Dataset. Like that dataset, we divide the area into 54 zones. A demonstration of this is shown in Figure 10. Since we do not have any rideshare or taxicab dataset, we generate the data randomly. driver and passenger origin and destination were generated randomly. Similarly, driver journey start time and passenger pickup start time were also generated randomly. For the rural dataset, we assume the passenger's pickup time window is 30 minutes. Distance between all the nodes in the road network was obtained from the OpenStreetMap router API. All other parameters and their values used in the experiments are similar to the NYC dataset. They are summarized in Table 7. Since there is no historical data available, we do not use the prediction mechanism for the rural dataset. In the next section, we discuss the performance of the rideshare system in a rural area.

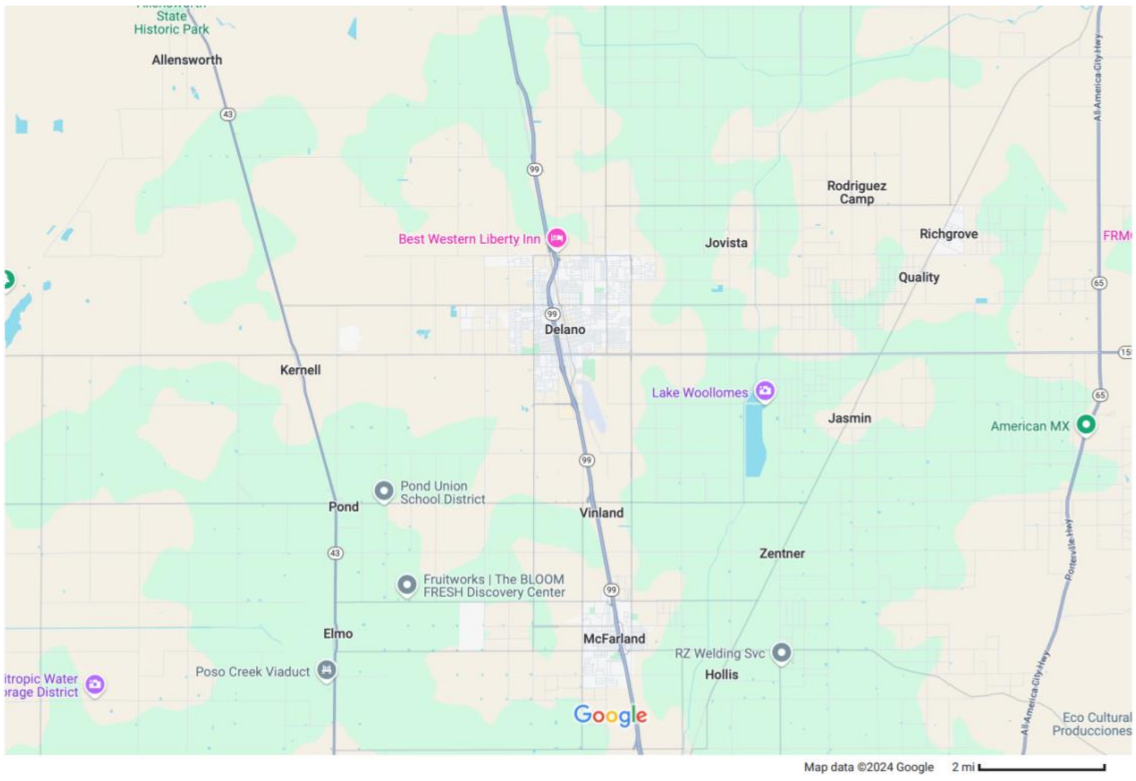


Figure 9. Map of the Rural Area

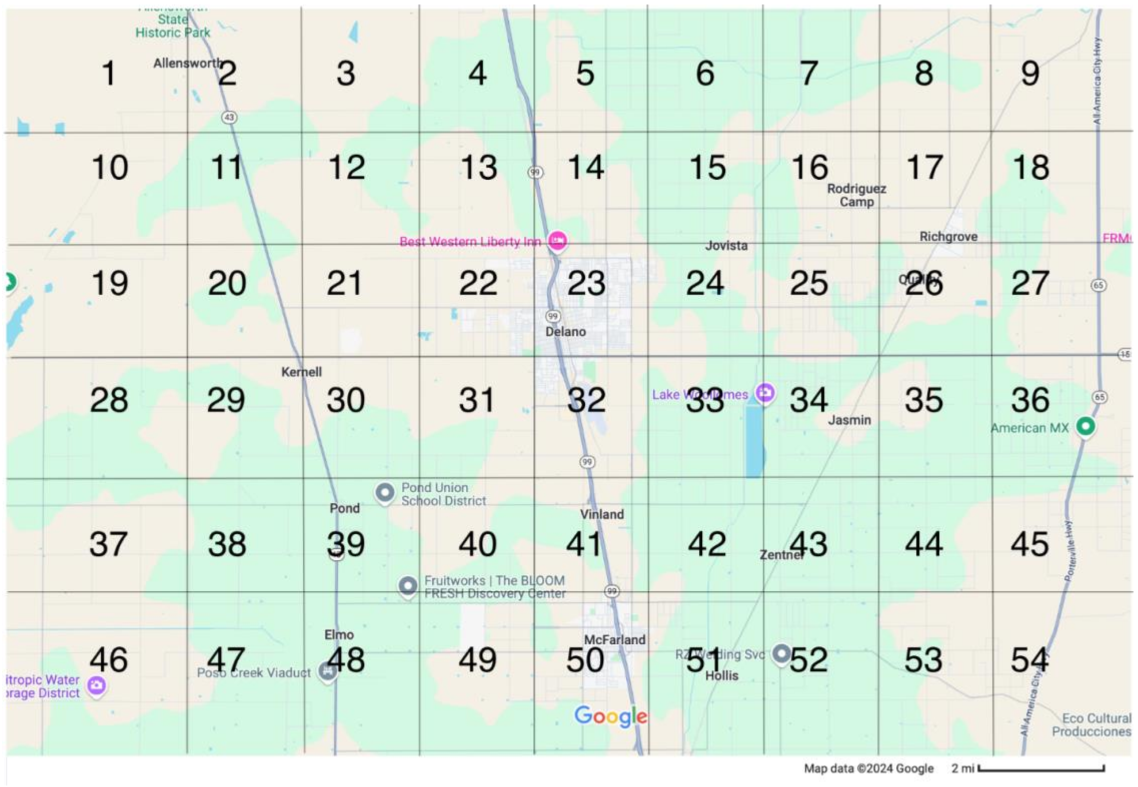


Figure 10. The Rural Dataset Broken Down into Zones

Table 7. Parameters and Their Values for the Rural Dataset

Parameter	Value
Planning Horizon, T	60 minutes
Planning time epoch, Δ	5 minutes
Pickup start time, a_p	$Uniform(0, T)$
Driver journey start time, a_d	$Uniform\ (0, T)$
Pickup time limit, b_p	Pickup start time, a_p+30 minutes
Hourly incentive budget, B	\$1000
Travel cost per km	\$0.187
Number of nearby drivers, k	20
Passenger walking limit, W_p	0.84 km
Driver ride time extension factor, ex_v	0.5
Passenger ride time extension factor, ex_p	0.5
Incentive payment per minute of maximum ride time violation, f_1 and f_2	\$1.00
Penalty for per kilometer distance increment, f_3	\$1.00
Upper limit of the benefit-incentive ratio, U	10
Lower limit of the benefit-incentive ratio, L	1

Performance of the Rideshare System in the Rural Dataset

For the rural dataset, we perform experiments on instances of smaller sizes than the NYC dataset to investigate the performance of the ridesharing system in an area with a sparse population. Both the number of drivers and the number of passengers ranges from 100 to 1,000. We report the same set of output parameters as for the NYC dataset.

Table 8 shows the simulation output for the rural area. Similar to Table 2, we compare the total vehicle distances traveled in two scenarios: solo driving and ridesharing across all test instances. From the Table 8, we can see that ridesharing can reduce the total vehicle distances traveled by 31%. From this discussion, we can say that ridesharing can significantly reduce the total vehicle miles traveled.

Table 9 shows the passenger service rate, the budget usage rate, the average occupancy per vehicle, and the average distance traveled by the rideshare vehicles. As we can see from the Table 9, the average passenger service rate is 79% compared to 33% in the NYC dataset. The difference is due to the smaller scale of the rideshare system. Although the incentive budget is 50% of the Manhattan experiments, due to the smaller number of requests per hour, more incentives can be paid, resulting in a higher passenger service rate. However, a decreasing trend in passenger service rate can be seen with the increasing size of test instances. Since the incentive budget is the same for all test instances, as the number of drivers and passengers increases, a lesser percentage of participants can be incentivized, as can be seen in the 4th column of Table 10, resulting in a lower passenger service rate. This is also reflected in the budget usage rate, which increases with increasing sizes of the problem instances and goes to 100% for larger instances. The average occupancy rate also decreases with increasing test instance sizes. This is again due to the decreasing passenger service rate. The geographic sparsity of the rural dataset is reflected in the average trip distance of rideshare vehicles. For the rural dataset, the average trip distance is 25.50 kilometers, more than double the average trip distance for the NYC dataset.

Table 8. Total Vehicle Distance Traveled across All Instances in the Rural Dataset

No. Drivers	No. Passengers	Total Vehicle Distance Traveled in Solo Driving (KM)	Total Vehicle Distance Traveled in Ridesharing (KM)	Improvement (%)
100	100	3770.62	2884.25	24%
200	200	7622.68	5809.50	24%
300	300	11434.61	8325.27	27%
400	400	14928.65	10693.06	28%
500	500	18848.31	13049.27	31%
600	600	22803.69	14962.32	34%
700	700	26536.56	16792.47	37%
800	800	30055.35	19600.68	35%
900	900	33915.53	22217.25	34%
1000	1000	37919.08	25711.51	32%
Average		**20783.51**	**14004.56**	**31%**

Table 9. Passenger Service Rate, Budget Usage Rate, Average Occupancy Rate, and Average Trip Distance per Rideshare Vehicle Across All Instances in the Rural Dataset

No. Drivers	No. Passengers	Passenger Service Rate (%)	Percent Budget Used	Avg. Occupancy per Vehicle	Trip Distance per Vehicle (km)
100	100	100%	22.91%	2.00	28.84
200	200	100%	47.21%	2.00	29.05
300	300	99%	69.71%	1.99	27.67
400	400	99%	85.22%	1.99	26.73
500	500	87%	100.00%	1.87	26.02
600	600	76%	100.00%	1.76	24.77
700	700	70%	100.00%	1.70	23.85
800	800	57%	100.00%	1.57	23.01
900	900	55%	100.00%	1.55	22.69
1000	1000	49%	100.00%	1.49	22.34
Average		**79%**	**82.51%**	**1.79**	**25.50**

Table 10 shows the average shared cost of the passengers, the percentage of participants receiving incentives, and the average incentive payment. The average shared cost of the passengers is $3.51, which is double the shared cost of the NYC dataset. This increase is due to spatial sparsity of the origin and destination of the passengers and drivers.

However, the shared cost is still significantly lower than ride-hailing services. The percentage of the participants receiving incentive payment decreases as the size of the test instances increases. This is due to the decreasing passenger service rate and the fixed incentive budget. The average incentive payment is $4.87. From this discussion, we can conclude that ridesharing can be beneficial in rural areas by providing cheap and flexible commutes to people where public transport is not available, and ride-hailing services are either not available or costly. The passenger service rate is also high for the rural case, although the number of rideshare participants is lower than in an urban area.

Table 10. Average Shared Cost, Incentive Recipient Rate, and the Average Incentive Payment Across All Instances in the Rural Dataset

No. Drivers	No. Passengers	Avg. Shared Cost ($)	Percent Participants Receiving Incentive	Avg. Incentive Payment ($)
100	100	3.59	45%	4.65
200	200	3.70	42%	5.31
300	300	3.57	47%	4.99
400	400	3.38	45%	4.73
500	500	3.39	39%	5.04
600	600	3.56	34%	4.82
700	700	3.52	31%	4.65
800	800	3.50	24%	5.12
900	900	3.43	23%	4.69
1000	1000	3.50	21%	4.65
Average		**3.51**	**35%**	**4.87**

Conclusions

In this study, we developed a framework for a real-time large-scale online rideshare system where a user can request a ride at any time, or a driver can enter or exit the system anytime. We developed a greedy heuristic for routing, a meeting point selection mechanism for flexible pickup or drop-off point selection, and a prediction mechanism for forecasting future ride requests and repositioning drivers. We also proposed an incentive system that incentivizes more people to participate in the rideshare system. We also integrated simulation software into our routing mechanism, which can simulate the rideshare routes and other traffic and provide the most up-to-date information about the travel time and traffic conditions of a road network. We performed numerical experiments on two datasets: one in New York City and one in Tulare and Kern Counties, California. Results from the numerical experiments show that ridesharing can be an effective way of providing flexible commutes to people at a fraction of the cost of a taxicab or ride-hailing service. Rideshare can also help improve the traffic congestion in an area, as can be seen from the simulation output. Also, our proposed incentive system can be extremely helpful in increasing the adoption of rideshare. As the results show, as the budget for incentives increases, the passenger service rate also increases. Our proposed framework can solve the driver-passenger matching and driver-routing problem in a city-scale network with 10,000 requests per hour. Thus, our framework is highly scalable.

Our proposed framework, numerical experiments, and results can be helpful to city officials. They can use our framework to launch a rideshare service that is incentivized by the government to reduce traffic congestion as well as increase accessibility to cheap transportation. Also, transportation planners can use our incentive mechanism to promote carpool ridesharing. Results from the simulation can help transportation planners understand the benefits of the rideshare system. This framework can also be applied to other major cities such as Los Angeles, Chicago, or San Francisco, where public transportation is insufficient to serve the large population.

Future research can look into developing routing algorithms based on exact solution procedures. Although our proposed algorithm can route hundreds of requests within seconds, it is based on a greedy heuristic. Exact methodologies that are scalable can provide even more distance savings and serve more passengers. Another direction of future work can be parallel matching algorithms. Although these parallel matching systems have been implemented for ride-hailing services, for ridesharing services these algorithms have not been explored. Future research can investigate parallelly running the simulation mechanism alongside the routing mechanism. This will improve the computational efficiency of the system. The system can update the travel times whenever the simulation module finishes.

References

Agatz, N., Erera, A. L., Savelsbergh, M. W. P., & Wang, X. (2011). Dynamic Ride-Sharing: A Simulation Study in Metro Atlanta. *Procedia - Social and Behavioral Sciences*, *17*, 532–550. https://doi.org/10.1016/j.sbspro.2011.04.530

Alonso-Mora, J., Samaranayake, S., Wallar, A., Frazzoli, E., & Rus, D. (2017). On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment. *Proceedings of the National Academy of Sciences*, *114*(3), 462–467. https://doi.org/10.1073/pnas.1611675114

Amrouss, A., El Hachemi, N., Gendreau, M., & Gendron, B. (2017). Real-time management of transportation disruptions in forestry. *Computers & Operations Research*, *83*, 95–105. https://doi.org/10.1016/j.cor.2017.02.008

Bertsimas, D., Jaillet, P., & Martin, S. (2019). Online Vehicle Routing: The Edge of Optimization in Large-Scale Applications. *Operations Research*, *67*(1), 143–162. https://doi.org/10.1287/opre.2018.1763

Chakrabarty, D., Zhou, Y., & Lukose, R. (2008). Online knapsack problems. *Workshop on Internet and Network Economics (WINE)*, 1–9. http://www.cs.dartmouth.edu/~deepc/PUBS/CZL-full.pdf

Chen, P., Ioannou, P., & Dessouky, M. (2021). Mixed freight dynamic routing using a co-simulation optimization approach. *IEEE Transactions on Intelligent Transportation Systems*, *23*(8), 12833–12845.

Cipolina-Kun, L. (2023). Enhancing Smart, Sustainable Mobility with Game Theory and Multi-Agent Reinforcement Learning With Applications to Ridesharing. *Proceedings of the AAAI Conference on Artificial Intelligence*, *37*(13), 16113–16114. https://ojs.aaai.org/index.php/AAAI/article/view/26917

Cui, Z., Henrickson, K., Ke, R., & Wang, Y. (2020). Traffic Graph Convolutional Recurrent Neural Network: A Deep Learning Framework for Network-Scale Traffic Learning and Forecasting. *IEEE Transactions on Intelligent Transportation Systems*, *21*(11), 4883–4894. https://doi.org/10.1109/TITS.2019.2950416

Dessouky, M. (2021). *Dynamic Routing for Ride-Sharing*. https://doi.org/10.7922/G2D798QK

Ferrucci, F., & Bock, S. (2015). A general approach for controlling vehicle en-route diversions in dynamic vehicle routing problems. *Transportation Research Part B: Methodological*, *77*, 76–87. https://doi.org/10.1016/j.trb.2015.03.003

Fielbaum, A. (2022). Optimizing a vehicle’s route in an on-demand ridesharing system in which users might walk. *Journal of Intelligent Transportation Systems*, *26*(4), 432–447. https://doi.org/10.1080/15472450.2021.1901225

Fielbaum, A., Bai, X., & Alonso-Mora, J. (2021). On-demand ridesharing with optimized pick-up and drop-off walking locations. *Transportation Research Part C: Emerging Technologies*, *126*, 103061. https://doi.org/10.1016/j.trc.2021.103061

Fikar, C. (2018). A decision support system to investigate food losses in e-grocery deliveries. *Computers & Industrial Engineering*, *117*, 282–290. https://doi.org/10.1016/j.cie.2018.02.014

Furuhata, M., Daniel, K., Koenig, S., Ordonez, F., Dessouky, M., Brunet, M.-E., Cohen, L., & Wang, X. (2014). Online Cost-Sharing Mechanism Design for Demand-Responsive Transport Systems. *IEEE Transactions on Intelligent Transportation Systems*, 1–16. https://doi.org/10.1109/TITS.2014.2336212

Hu, S., Dessouky, M. M., Uhan, N. A., & Vayanos, P. (2021). Cost-sharing mechanism design for ride-sharing. *Transportation Research Part B: Methodological*, *150*, 410–434.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. *Advances in Neural Information Processing Systems*, *30*. https://proceedings.neurips.cc/paper/2017/hash/6449f44a102fde848669bdd9eb6b76fa-Abstract.html

Krajzewicz, D. (2010). Traffic Simulation with SUMO – Simulation of Urban Mobility. In J. Barceló (Ed.), *Fundamentals of Traffic Simulation* (Vol. 145, pp. 269–293). Springer New York. https://doi.org/10.1007/978-1-4419-6142-6_7

Kumar, A., Vorobeychik, Y., & Yeoh, W. (2023). *Using Simple Incentives to Improve Two-Sided Fairness in Ridesharing Systems*. https://doi.org/10.48550/ARXIV.2303.14332

Lasley, P. (2021). *2021 URBAN MOBILITY REPORT*. https://mobility.tamu.edu/umr/report/

Li, X., Hu, S., Fan, W., & Deng, K. (2018). Modeling an enhanced ridesharing system with meet points and time windows. *PLOS ONE*, *13*(5), e0195927. https://doi.org/10.1371/journal.pone.0195927

Li, Z., Liang, C., Hong, Y., & Zhang, Z. (2021). How Do On-demand Ridesharing Services Affect Traffic Congestion? The Moderating Role of Urban Compactness. *Production and Operations Management*, poms.13530. https://doi.org/10.1111/poms.13530

Lowalekar, M., Varakantham, P., & Jaillet, P. (2021). Zone pAth Construction (ZAC) based Approaches for Effective Real-Time Ridesharing. *Journal of Artificial Intelligence Research*, *70*, 119–167. https://doi.org/10.1613/jair.1.11998

Ma, S., Zheng, Y., & Wolfson, O. (2015). Real-Time City-Scale Taxi Ridesharing. *IEEE Transactions on Knowledge and Data Engineering*, *27*(7), 1782–1795. https://doi.org/10.1109/TKDE.2014.2334313

Monroy-Licht, M., Amaya, C. A., Langevin, A., & Rousseau, L. (2017). The rescheduling arc routing problem. *International Transactions in Operational Research*, *24*(6), 1325–1346. https://doi.org/10.1111/itor.12346

New York (N.Y.). Taxi And Limousine Commission. (2019). *New York City Taxi Trip Data, 2009-2018: Version 1* (Version v1) [Dataset]. ICPSR - Interuniversity Consortium for Political and Social Research. https://doi.org/10.3886/ICPSR37254.V1

Novaes, A. G. N., Bez, E. T., Burin, P. J., & Aragão, D. P. (2015). Dynamic milk-run OEM operations in over-congested traffic conditions. *Computers & Industrial Engineering*, *88*, 326–340. https://doi.org/10.1016/j.cie.2015.07.010

Ojeda Rios, B. H., Xavier, E. C., Miyazawa, F. K., Amorim, P., Curcio, E., & Santos, M. J. (2021). Recent dynamic vehicle routing problems: A survey. *Computers & Industrial Engineering*, *160*, 107604. https://doi.org/10.1016/j.cie.2021.107604

Shen, J., Tziritas, N., & Theodoropoulos, G. (2022). A Baselined Gated Attention Recurrent Network for Request Prediction in Ridesharing. *IEEE Access*, *10*, 86423–86434. https://doi.org/10.1109/ACCESS.2022.3199423

Simonetto, A., Monteil, J., & Gambella, C. (2019). Real-time city-scale ridesharing via linear assignment problems. *Transportation Research Part C: Emerging Technologies*, *101*, 208–232. https://doi.org/10.1016/j.trc.2019.01.019

Tafreshian, A., & Masoud, N. (2022). A Traveler Incentive Program for Promoting Community-Based Ridesharing. *Transportation Science*, *56*(4), 827–847. https://doi.org/10.1287/trsc.2021.1121

Ulmer, M. (2017). Delivery deadlines in same-day delivery. *Logistics Research*, *10*(3), 1–15.

Ulmer, M. W., Thomas, B. W., Campbell, A. M., & Woyak, N. (2021). The Restaurant Meal Delivery Problem: Dynamic Pickup and Delivery with Deadlines and Random Ready Times. *Transportation Science*, *55*(1), 75–100. https://doi.org/10.1287/trsc.2020.1000

Wang, T., Luo, H., Bao, Z., & Duan, L. (2023). Dynamic Ridesharing With Minimal Regret: Towards an Enhanced Engagement Among Three Stakeholders. *IEEE Transactions on Knowledge and Data Engineering*, *35*(4), 3712–3726. https://doi.org/10.1109/TKDE.2022.3141368

Wang, Y., Yin, H., Chen, T., Liu, C., Wang, B., Wo, T., & Xu, J. (2022). Passenger Mobility Prediction via Representation Learning for Dynamic Directed and Weighted Graphs. *ACM Transactions on Intelligent Systems and Technology*, *13*(1), 1–25. https://doi.org/10.1145/3446344

Xu, Y., Kulik, L., Borovica-Gajic, R., Aldwyish, A., & Qi, J. (2020). Highly Efficient and Scalable Multi-hop Ride-sharing. *Proceedings of the 28th International Conference on Advances in Geographic Information Systems*, 215–226. https://doi.org/10.1145/3397536.3422235

Zhao, Y., Ioannou, P. A., & Dessouky, M. M. (2018). Dynamic multimodal freight routing using a co-simulation optimization approach. *IEEE Transactions on Intelligent Transportation Systems*, *20*(7), 2657–2667.

Data Summary

Products of Research

One of the main research products of this research will be peer-reviewed journal articles, book chapters, and/or conference proceedings targeted toward the transportation science research community, plus supplemental materials such as tables, numerical data used for graphs, etc. The resulting algorithms will be published in peer-reviewed journals.

Data Format and Content

All research products will be available online in digital form. Manuscripts will appear in a common document-viewing format, such as PDF, and supplemental materials, such as tables and numerical data, will be in a tabular format, such as Microsoft Excel spreadsheet, tab-delimited text, etc. The New York Taxicab Dataset is found in PARQUET format.

Data Access and Sharing

All participants in the project will publish the results of their work. Papers will be published in peer-reviewed scientific journals, books published in English, conference proceedings, or as peer-reviewed data reports. Beyond the data posted on USC websites, primary data and other supporting materials created or gathered in the course of work will be shared with other researchers upon reasonable request, at no more than incremental cost and within a reasonable time of the request or, if later, the filing of a patent application covering the results of such research.

All the data used in this research report can be found at Figshare: https://dx.doi.org/10.6084/m9.figshare.28119947. This includes the maps in the XML format, distance data, coordinates of the zones, all the data in the tables, and the data for the graphs. The New York Taxicab and Limousine Commission Dataset can be found at: https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page

Reuse and Redistribution

USC's policy is to encourage, wherever appropriate, research data to be shared with the public through internet access. The university will regulate this public access to protect privacy and confidentiality concerns, as well as to respect any proprietary or intellectual property rights. Administrators will consult with the university's legal office to address any concerns on a case-by-case basis, if necessary. Terms of use will include requirements of attribution along with liability disclaimers in connection with any use or distribution of the research data, which may be conditioned under some circumstances.